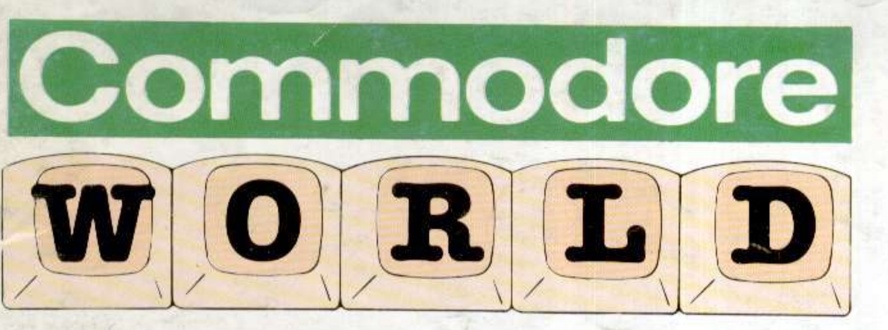


Nº 8 - OCTUBRE 1984

315 PTAS.

PARTICIPA
CERCA DE 2.500.000 PESETAS, EN PREMIOS DEL 1.er CONCURSO NACIONAL ESCOLAR DE PROGRAMACION (VER PAG. 13)

Viaje al centro de tu Commodore

Qué hacer con tu datassette para que entren esas cintas "rebeldes"
No te deshagas de tu VIC cuando compres un C-64
El VIC-20 por dentro
Pasaporte al mundo
Disk-0-64

¡POR FIN!

Ha llegado un Nº 1 en USA para su

Commodore 64

SAUCER ATTACK!

An action game for the Commodore 64™

Joystick required

Ud. está en su posición vigilando Washington D.C.

La ciudad parece apacible desde su puesto de defensa y no hay los signos de la invasión Extraterrestre que Ud. espera.

De pronto el silencio es alterado por el penetrante silbido de los platillos volantes. Son los invasores descendiendo sobre la ciudad.

Ud. entra en acción y pone en su punto de mira al platillo volante.

Aprieta el botón de juego y sale un misil que destruye al ovni desintegrándose el platillo. Otro platillo volante se acerca a los edificios y antes de que Ud. se de cuenta destruye el monumento de Washington C.D. con un rayo de energía atómica. La batalla transcurre durante el día y en un momento determinado pasa a ser de noche.

Si un platillo aterriza todo está perdido. Cuando Ud. consiga neutralizar el ataque el juego continuará y se deberá destruir la nave madre. Si lo consigue una gran celebración le espera cuando vuelva a Washington D.C.

SAUCER ATTACK
es realmente
***un* JUEGO DISTINTO**





FOTOS TOMADAS DIRECTAMENTE DE UN MONITOR 1702 COMMODORE

INSUPERABLE CALIDAD AUDIO-VISUAL

PEDIDOS

FERRE-MORET S.A.

Tuset n.º 8, entlo. 2.º Tel. 218 02 93
BARCELONA 08006

DISTRIBUIDORES Y PARTICULARES

NOMBRE...
DIRECCION.......................................
CIUDAD.................... PROVINCIA ....

☐ Adjunto cheque
☐ Contra reembolso

Gastos envio: **300 ptas.**

Precio Venta **9.975 ptas.**

SUPER-INTERESANTISIMO

CERCA DE 2.500.000 PTAS. EN PREMIOS

1.er CONCURSO NACIONAL ESCOLAR DE PROGRAMACION (bases pág. 5, premios pág. 13)

Ya tenemos la relación de premios que se otorgarán a los futuros ganadores del 1.er CONCURSO NACIONAL ESCOLAR DE PROGRAMACION.

NIVEL EGG.—100.000 pesetas en material a elegir al concursante individual o equipo. El centro representado por el ganador recibirá, a elegir, un Ordenador SX-64 ó bien 5 Ordenadores Commodore del nuevo modelo 16 con sus correspondientes Cassettes.

NIVELES F.P. y BUP.—150.000 pesetas en material a elegir para cada uno de los ganadores (individual o equipo) de estas dos categorías. Los Centros representados por los ganadores recibirán, a elegir, un Ordenador Profesional Commodore con sus unidades de disco o bien 5 Commodore 16 con sus correspondientes Cassettes. Todos los ganadores, incluidos los centros, recibirán asimismo suscripciones a Commodore World y Micro-Sistemas.

Ya sólo queda el animaros a todos, centros y alumnos a darle al teclado, a la imaginación y a la creatividad.

REPETICION DE PROGRAMAS DE NUESTRO Nº 7

Parece ser que en un momento dado, parte de la tirada de nuestro número 7, correspondiente a septiembre salió con varios listados defectuosos debido a algún fallo mecánico de la imprenta. En páginas centrales de este número aparece una separata extra con repetición de estos listados, ya que parece haber sido bastante alto el número de lectores afectados. Lamentamos este incidente, completamente ajeno a nosotros, y damos las gracias a Ibergráficas por haberse ofrecido inmediatamente a la impresión de esta separata.

NUMERO ESPECIAL DE NAVIDAD

Estamos preparando un volumen especial de programas (de 75 a 100) cara a las fiestas de Navidad. Este número especial, que esperamos poder presentar en SIMO, lo podrán adquirir nuestros suscriptores a precio de coste y el resto de nuestros lectores con un descuento del 25%. El precio de venta al público será de 1.250 pesetas. Por lo tanto el precio de nuestros suscriptores será de 625 pesetas y a los lectores no suscriptores de 935 pesetas.

La tirada tendrá un número limitado por lo que ya podéis reservar la copia o copias que deseéis (boletín pie de página) pues no habrá reimpresión una vez que se agoten.

EL EQUIPO AUMENTA

Diego ya tiene ayudante, Alvaro Ibáñez, que a sus 16 años y estudiante de COU es todo un joven genio del Computador. Bienvenido en nombre de todos (¡y especialmente de Diego!).

SUSCRIPTORES Y SEÑAS CORRECTAS

Tras haberse metido todas las suscripciones en el ordenador, si hay algún lector que observe alguna incorrección en su nombre o dirección, le rogamos nos lo comunique, así como cualquier cambio de domicilio que se haya producido, a fin de evitar retrasos, pérdidas o devoluciones.

Y hablando de suscriptores... a por el 8.000... y... después del descanso veraniego, ya nos están llegando otra vez a toda marcha.

Estamos en el suscriptor 5.670. Cuando lleguemos al 8.000 se sortearán los viajes que se indican en la página 5 (Concursos Permanentes).

Y HABLANDO DE VIAJES

Sobre el viaje a Londres nada podemos deciros de momento ya que nuestro ganador, todavía no ha tenido esa semana libre para poder ir. Sin embargo, Diego y Joan se largaron a Estados Unidos, como estaba previsto, y parece ser que no lo pasaron nada mal. Lo que Diego nos cuenta sobre la Fábrica de Circuitos Integrados de Commodore es fascinante. Pero mejor dejo que él haga un resumen de sus experiencias.

PASEANDO POR "LAS AMERICAS"

El día 19 de julio Joan Pujiula i Junca llegó a la estación de Chamartín en Madrid, le dejamos refrescarse, y le enseñamos las oficinas y luego toda la capital en un rápido tour en coche.

El día 20 por la mañana temprano, salíamos de Madrid con destino a Nueva York. ¿Qué aventuras viajeras nos esperaban?

Nos recogió Doris Moris, del Instituto de Asuntos Culturales (ICA), que nos invitó a una cena-merienda-almuerzo (según el reloj con que se mire) y nos enseñó parte de la ciudad.

El día siguiente tuvimos libertad para recorrer tiendas, ver parte de la ciudad y descansar, aunque el tiempo se puso en contra, lluvia torrencial toda la mañana, no nos logró desanimar, compramos un paraguas de los que venden a 3$ en todas las esquinas de Nueva York los días de lluvia y seguimos.

Esa tarde amainó el temporal y fuimos con otros dos jóvenes de ICA a ver el espectáculo del Radio City Hall, fue fabuloso, tanto de colorido visual como musical. La sonorización de la sala y los medios materiales (elevadores, escenarios, proyectores, etc.), lo podéis imaginar, era de lo mejor, para eso se trata de uno de los edificios del centro Rockefeller.

El domingo por la mañana hicimos el tour del alto y bajo Manhatan al que nos invitó Ibercharter, como podréis imaginar recorrimos todo, desde el parque de las baterías (al sur) hasta casi el Bronx (al norte), y desde Brodway-oeste y las torres gemelas al edificio de Naciones Unidas al este, es curioso, pero ese día el sol estaba tan luminoso que vendían gafas de sol en todas las esquinas de la ciudad también a 3 dólares.

El lunes por la mañana tomamos un avión desde el aeropuerto Kennedy hasta el de Filadelfia, donde tomamos una limusine hasta el hotel Sheraton de Valley Forge, que tienen unas vistas preciosas del valle y todo tipo de instalaciones deportivas y de descanso, nosotros optamos por la segunda (lógico, ¿no?), y una cena estilo europeo (estábamos cansados de comida americana).

El martes temprano nos recogió la limusine matrícula "CBM-1", se trata del coche de los jefazos de Commodore. Es un Cadillac larguísimo con bar, televisión, y todo dentro. Nos llevó a la planta de West Chester donde fabrican en la actualidad una cifra escalofriante de millones de dólares en commodores 64. Creo que me bloqueé la cantidad a pesetas y perdí la mitad de la frase. Nos enseñaron toda la fábrica, desde el lugar donde se recibe la materia prima al lugar donde se envían en camiones los C-64 a sus destinos, pasando por toda la cadena de producción.

Fuimos a comer con Mr. Sullivan y Mr. Kelly, que nos llevaron a dar una vuelta en la limusine y luego le dijeron al chófer que nos llevase a la fábrica de circuitos integrados de MOS Tecnology, donde nos esperaba el señor James G. Heil (gerente de seguridad y asesor de agencias de protección de entornos en Commodore Semiconductor Systems), vimos las salas de diseño y proyecto donde se "cuecen" las ideas que luego se convierten en chips, vimos la zona de producción desde unas ventanas, nos explicaron lo que hacía cada señor y por fin pasamos a unas salas que, como las salas de descompresión en los buzos o las antecámaras de depresión de las centrales nucleares, nos preparamos para pasar a la zona aséptica, tuvimos que ponernos unas zapatillas especiales, unos gorritos para el pelo, el bigote y las cejas, y unas batas protectoras. La razón de todo esto, es que basta una pequeña partícula de polvo para que un chip no sirva, igual mente durante la fabricación es suficiente una pequeña carga estática para destruir el más complejo de los circuitos integrados, y cada Waffer (base sobre la que se van depositando todas las capas que forman el circuito integrado), tarda unos 24 días en fabricarse, con lo

(Pasa a la pág. 66)

BOLETIN DE RESERVA ESPECIAL NAVIDAD

NOMBRE .... EDAD ....

DIRECCION .... POBLACION ....

PROVINCIA .... TELEFONO ....

Nº Suscriptor [ ] ENVIO TALON ☐ GIRO ☐

No soy suscriptor pero adjunto el boletín de suscripción (pág. 7) y me acojo al precio especial para suscriptores.... ☐

ª Precio: suscriptor 625 pesetas. Lector no suscriptor 935 pesetas.

Convocatoria del Primer Concurso Nacional Escolar de Programación patrocinado por Commodore World y Microelectrónica y Control

BASES

1º Podrán participar en el mismo todos aquellos niños y jóvenes que estén en edad escolar pre-universitaria o equivalente.

2º Habrá tres grupos definidos:
a) E.G.B.
b) B.U.P.
c) F.P.

3º Los trabajos pueden ser presentados por equipos o por individuales.

4º Los trabajos presentados por equipos deben especificar:
a) Nombre del equipo.
b) Centro al que representan.
c) Nivel de enseñanza que cursan (E.G.B.-B.U.P-F.P.).
d) Nombre del profesor, monitor o coordinador.

5º Los trabajos presentados por individuales deben ir acompañados de:
a) Nombre del autor.
b) Edad y curso que estudia.
c) (¡) Si concursa representando a un centro: nombre del mismo y nombre del profesor responsable.
(¡¡) Si concursa por sí mismo: nombre del colegio donde estudia y firma del director o profesor responsable avalando que en tanto cuanto conoce al concursante el trabajo que presenta es suyo propio.

6º El tema del concurso es la realización de un programa educativo sobre materia libre a elegir por los concursantes. La extensión del programa es también de libre elección.

7º Cada centro puede presentar el número de equipos o concursantes individuales que desee, siempre que cada trabajo sea enviado por separado.

8º Los trabajos deben incluir:
a) Nombre el programa.
b) Nombre del computador para el que está confeccionado y periféricos adicionales que se necesitan (si fuese el caso).
c) Objetivo del programa.
d) Explicación detallada del programa (a máquina o impresora).
e) Listado completo a impresora.
f) Grabación en cinta o disco.

9º Todos los trabajos presentados deben ser originales e inéditos.

10º Los centros que pueden presentar equipos o concursantes individuales son:
a) Todo colegio, escuela o instituto, público o privado que imparta las enseñanzas de E.G.B., B.U.P, o F.P.
b) Centros de Informática que tengan grupos de enseñanza específica para niños y jóvenes menores de 18 años.
c) Clubs de informática que mantengan actividades específicas para niños y jóvenes menores de 18 años.

11º Los trabajos deben ser enviados antes del 31 de diciembre de 1984 a SIMSA, c/ Barquillo, 21-3º Izqda. 28004 Madrid.

12º El fallo del concurso se dará a conocer en la primera semana de marzo de 1985 en el lugar, fecha y hora que se **indicarán**.

13º Los trabajos premiados quedarán propiedad de los patrocinadores.

RELACION DE PREMIOS EN PAG. 13

CONCURSOS PERMANENTES

Colaboraciones de Programas y Magia

La verdad sea dicha: os estáis portando; las colaboraciones que nos están llegando son numerosas y de muy buena calidad. Por lo tanto, vamos a aclarar las condiciones de nuestros concursos permanentes.

1) Los sorteos se realizarán en los meses de junio y diciembre.

2) Se sortearán siempre seis premios, tres de 15.000, 10.000 y 5.000 pesetas en metálico de "Commodore World" y tres del mismo valor en material de Microelectrónica y Control, entre todas las colaboraciones publicadas.

3) Se sortearán siempre seis premios, tres de 3.000, 2.000 y 1.000 pesetas en metálico de "Commodore World" y tres del mismo valor en material de Microelectrónica y Control, entre todas las contribuciones publicadas en la sección de "Magia".

4) Adicionalmente, se sortearán diversos premios de cintas, juegos, etc. donados por firmas y distribuciones, entre todas las colaboraciones.

5) Los autores de las colaboraciones vendidas dentro de nuestro "Servicio de Cintas" percibirán el 20% del precio de la cinta.

6) Todas las colaboraciones deben venir escritas a máquina y los programas grabados en cinta (si es posible), o con el listado completo en impresora. Nuestros lectores más jóvenes pueden escribir a mano pero con letra muy clara.

7) Quedan automáticamente descalificadas tanto del sorteo como del "Servicio de Cintas" las colaboraciones que hayan sido enviadas a otras revistas.

8) Las colaboraciones se enviarán a "Commodore World", *C/Barquillo, 21-3 izda. 28004 Madrid.*

MAS VIAJES Y OTRO REGALO DE "IMPRESION"... A POR EL SUSCRIPTOR 8.000

Es evidente que a nuestros sucriptores les gusta viajar. De momento Joan Pujiula visitará los Estados Unidos en julio y Juan Mejuto Iglesias irá a Inglaterra en el otoño. Queríamos sortear algo un poco especial ENTRE TODOS LOS SUCRIPTORES cuando lleguemos al suscriptor número 8.000, y hemos optado por dos premios, uno concedido por Microelectrónica y Control y otro concedido por "Commodore World".

MEC regalará una impresora MPS 801 (o similar), y nosotros hemos escogido un surtido de viajes muy "chulos" para dos personas para que el ganador elija el que más le apetezca. La selección que hemos hecho es la siguiente:

Viajes de 4 días en hoteles de primera clase a uno de estos lugares: París, Lisboa, Atenas, Amsterdam, Niza, Viena.
O viajes de una semana en hoteles de 4/5 estrellas: Canarias o Mallorca.

NO OLVIDEIS
NUESTRA NUEVA
DIRECCION:
BARQUILLO, 21-3º IZDA.
28004 MADRID
TELEF.: 231 23 88
231 23 95

TIRADA DE COMMODORE WORLD
Certificación de 28.000 ejemplares

Según certificado firmado por la auditoría internacional DELOITTE HASKINS SELLS, S. A. el 5 de marzo de 1984, la tirada de Commodore World es de 28.000 ejemplares. Este certificado se encuentra en las oficinas de S.I.M.S.A. para comprobación de cualquier persona o entidad que lo desee.

Convocatoria del primer concurso nacional de programación en 80 columnas de Commodore-64 y 40-80 columnas del VIC-20 patrocinado por Ferre-Moret, S.A. Con la colaboración de Commodore-World, más de 500.000 ptas. en premios

BASES

1º Podrán participar todos los usuarios de COMMODORE-64 y VIC-20 que hayan obtenido la tarjeta de 80 columnas de FERRE-MORET, S.A.
2º Cada concursante podrá presentar, un programa realizado por él o por un equipo del que deberán indicar un titular.
3º Se enviará la explicación detallada del programa redactado en español, escrito a máquina. El programa se enviará en cassette o disco, adjuntando su correspondiente listado del programa.
4º Se enviarán por correo certificado a la siguiente dirección, FERRE-MORET, S.A. calle Buenos Aires nº 30, 2º-3.ª de BARCELONA-36. Antes del 31 de enero de 1985.
5º Los programas no premiados podrán ser solicitados por sus propietarios en el plazo de treinta días después del fallo.
6º El resultado se comunicará, directamente a los ganadores y se publicará en la revista COMMODORE WORLD.
7º Los premios se entregarán durante la primera semana de marzo de 1985 coincidiendo con la entrega de premios del PRIMER CONCURSO NACIONAL ESCOLAR DE PROGRAMACION.
8º Los programas premiados quedarán en propiedad de FERRE-MORET, S.A.
9º Los programas que no obtengan premio, recibirán, en caso que así lo considere FERRE-MORET, S.A., ofertas de compra de sus programas.
10º Los premios ascenderán a 500.000 Ptas. repartidos de la siguiente manera:

1º 250.000 Ptas.
2º 150.000 Ptas.
3º 100.000 Ptas.

11º Todos los participantes, que no hayan recibido premio u oferta de compra, podrán participar en el concurso de colaboradores que habitualmente COMMODORE WORLD viene realizando con un premio extra de FERRE-MORET, S.A., de 50.000 Ptas.
12º Los trabajos deberán de ser inéditos y desarrollar temas de utilidad, programas profesionales, científicos, educativos, etcétera.

FERRE-MORET, S.A. se reserva el derecho de modificar las bases de este certamen, dejarlo desierto o suprimirlo, por causa de fuerza mayor. En este caso la empresa se compromete a comunicarlo a través de COMMODORE WORLD.

Clave para interpretar los listados

Todos los listados que se publican en esta Revista han sido ejecutados en el modelo correspondiente de la gama de ordenadores COMMODORE. Para facilitar la edición de los mismos en la Revista y para mejorar su legibilidad por parte del usuario, se les ha sometido a ciertas modificaciones mediante un programa escrito especialmente para ello. Para los programas destinados a los ordenadores VIC-20 y COMMODORE 64, en los que se usan frecuentemente las posibilidades gráficas del teclado, se han sustituido los símbolos gráficos que aparecen normalmente en los listados por una serie de letras entre corchetes [] que indican la secuencia de teclas que se deben pulsar para obtener el carácter deseado. A continuación se da una tabla para aclarar la interpretación de las indicaciones entre corchetes:

[CRSRD] = Tecla cursor hacia abajo (sin SHIFT)

[CRSRU] = Tecla cursor hacia arriba con SHIFT)

[CRSRR] = Tecla cursor a la derecha (sin SHIFT)

[CRSRL] = Tecla cursor a la izquierda (con SHIFT)

[HOME] = Tecla CLR/HOME (sin SHIFT)

[CLR] = Tecla CLR/HOME (con SHIFT)

Las indicaciones [BLK] a [YEL] corresponden a la pulsación de las teclas de 1 a 8 junto a la tecla CTRL. Lo mismo sucede con [RVSON] y [RVSOF] respecto a la tecla CTRL y las teclas 9 y 10.

El resto de las indicaciones constan de la parte COMM o SHIF seguidas de una letra, número o símbolo —por ejemplo [COMM+] o [SHIFA]—. Esto indica que para obtener el gráfico necesario en el programa deben pulsarse simultáneamente las teclas COMMODORE (la que lleva el logotipo) o una de SHIFT y la tecla indicada por la letra, el número o el símbolo, en el ejemplo anterior: COMMODORE y + o SHIFT y A, respectivamente.

En los signos gráficos además se cuenta el número de veces que aparece. Por ejemplo, [7 CRSRR] equivale a 7 pulsaciones de la tecla cursor a la derecha y [3 SPC] tres pulsaciones de la barra espaciadora. ■

INDICE DE ANUNCIANTES

BOLETIN DE SUSCRIPCION — Commodore World

NOMBRE .... EDAD ....
DIRECCION ....
POBLACION .... (...........) PROVINCIA ....
TELEF. ...., MARCA Y MODELO DEL ORDENADOR ....
....
CIUDAD DONDE LO COMPRO .... DISTRIBUIDOR ....
APLICACIONES A LAS QUE PIENSA DESTINAR EL EQUIPO ....
....
....

Deseo iniciar la suscripción con el nº......... 9
Adjunto cheque de 2.530 pesetas.............. ☐
Reembolso más gastos del mismo
al recibir el primer nº de la suscripción..... ☐
(Enviar a la direccion del dorso)

Extranjero $40. Solamente por correo aéreo

Firma.

Desde 1º de julio

DESEO SUSCRIBIRME A **COMMODORE WORLD** POR UN AÑO AL PRECIO DE **2.530** PTS. DICHA SUSCRIPCION ME DA DERECHO, NO SOLO A RECIBIR LA REVISTA (ONCE NUMEROS ANUALES) SINO A PARTICIPAR EN LAS ACTIVIDADES QUE SE ORGANICEN EN TORNO A ELLA Y QUE PUEDEN SER COORDINACION DE CURSOS DE BASIC, INTERCAMBIOS DE PROGRAMAS, CONCURSOS, ETC.

EJEMPLARES ATRASADOS DE COMMODORE WORLD

Commodore WORLD

1	2	3	4	5	6	7

Precio por ejemplar 275 ptas. (Hasta el nº 5) a partir del nº 6 el precio es de 315 ptas. ejemplar
Forma de pago: sólo por cheque.

Peticionario ....
Calle .... **Nº** .... **TELEF.** ....
Población .... **D. P.** .... **Provincia** ....
Incluyo cheque por valor de .... **pesetas + 75 de gastos de envío.**

SERVICIO DE CINTAS

De programas aparecidos en Commodore World.

Titulo del programa .... publicado en nº ☐
Titulo del programa .... publicado en nº ☐
Titulo del programa .... publicado en nº ☐

Precio por cinta 850 pesetas.
* **Precio por disco 1.500 pesetas.**
} **Gastos de envío 75 pesetas. Forma de pago: sólo por cheque.**

Peticionario ....
Calle .... Nº .... TELEF. ....
Población .... D.P. .... Provincia ....

* *Programa evaluación de colegios sólo en discos.*
Incluyo cheque por .... **pesetas.**

Programa para VIC-20 ☐
Programa para C-64 ☐

EJEMPLARES ATRASADOS DE «CLUB COMMODORE»

Primera época (septiembre-1982 - enero-1984)

Para poder satisfacer la creciente demanda de Club Commodore, agotada en todos sus números, hemos puesto en marcha un Servicio para suministrar fotocopias de los ejemplares que nos sean solicitados.

SERVICIO DE FOTOCOPIAS.— NUMERO DE LA EDICION SOLICITADA.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Peticionario: D. ....
Calle .... Nº .... TELEF. ....
Población .... D.P. .... Provincia ....

Forma de pago sólo por cheque

Precio de la edición fotocopiada: 250 ptas.

La colección completa del 0 al 15: 2.500 ptas. + 150 ptas. por gastos de envío.
Incluyo cheque por .... **ptas.**

Commodore

W O R L D

Barquillo, 21, 3º Izda.
Teléf.: 231 23 85/86
28004 MADRID

Sant Gervasi de Cassoles, 39-despacho 4 o 4
Teléf.: 212 73 45 / 212 88 48
08022 BARCELONA

Commodore

W O R L D

Barquillo, 21, 3º Izda.
Teléf.: 231 23 85/86
28004 MADRID

Sant Gervasi de Cassoles, 39-despacho 4
Teléf.: 212 73 45 / 212 88 48
08022 BARCELONA

Commodore

W O R L D

Barquillo, 21, 3º Izda.
Teléf.: 231 23 85/86
28004 MADRID

Sant Gervasi de Cassoles, 39-despacho 4
Teléf.: 212 73 45 / 212 88 48
08022 BARCELONA

Commodore

W O R L D

Barquillo, 21, 3º Izda.
Teléf.: 231 23 85/86
28004 MADRID

Sant Gervasi de Cassoles, 39-despacho 4
Teléf.: 212 73 45 / 212 88 48
08022 BARCELONA

Karl T. THURBER (RUN EEUU)

Traducido y adaptado por Diego ROMERO

No te deshagas de tu VIC cuando compras un C-64

Por si no tienes suficiente con un solo ordenador



Este interface, de fácil construcción, te permitirá usar el VIC-20, el C-64, los periféricos y monitor con rapidez y facilidad

En el año 1982, compré un VIC-20 y unos meses más tarde un C-64, un monitor en color, dos unidades de disco 1541 y una impresora 1525. Fue fácil conectar las unidades de disco y la impresora a cualquiera de los dos ordenadores (pero no a los dos en un solo sistema) mediante el bus de serie I/O, dado que los dispositivos están conectados entre sí en forma de cadena.

Sin embargo, me interesaba seguir utilizando el VIC-20 con todos los periféricos ya que poseía una colección de software para el VIC (especialmente juegos) bastante completa. Además, los niños se divertían mucho con el VIC y les servía para aprender. Lo que me hacía falta era un interface capaz de hacer dos cosas: conmutar el bus de serie I/O de seis conexiones que llevaba los periféricos a cualquiera de los dos ordenadores, y conmutar las conexiones acústicas y de video al monitor en color.

Como no podía encontrar ningún interface hecho, lo tenía que construir yo mismo. El interface necesitaba un conmutador deslizante o rotativo de 2 circuitos, 2 posiciones para las líneas de àudio y video. (En vez de esto, se podía haber utilizado un solo conmutador de ocho circuitos).

Los conmutadores los monté en una pequeña caja de 8 × 5,5 × 10 cm., del tipo de laboratorios Multi o Retex. Instalé los conectores DIN hembra de seis patillas en el panel trasero, también monté en él dos jacks de tipo phono-RCA utilizados para las conexiones de audio y video.

Para conectar los buses serie de I/O de cada computador a la caja interface, empleé una manguera de seis cables no blindados con conectores machos DIN de seis patillas en los extremos. Para unir los conectores de salida de audio/video, preparé cables con conectores machos de tipo DIN de 5 patillas en un extremo y jacks-RCA machos en el otro. La unión del monitor a la caja la hice con un juego de cables de audio normales, y con otro cable de seis conductores conecté la caja del interface con el primer drive 1541 de la cadena de periféricos.

De este modo simple y sencillo, por menos de 3.000 pts., la pequeña caja-interface permite cambiar instantáneamente de un computador al otro, utilizando periféricos y monitor compartidos, sin necesidad de enchufar y desenchufar cables.

La única precaución a tener en cuenta, es apagar todo el equipo cuando se va a cambiar de ordenador, para evitar que voltajes transitorios de conmutación puedan dañar cualquier ordenador o periférico. Por supuesto los controles de color del monitor necesitarán un pequeño reajuste al pasar de un ordenador al otro. La figura 1 muestra el diagrama de bloques de las conexiones del interface descrito. ■

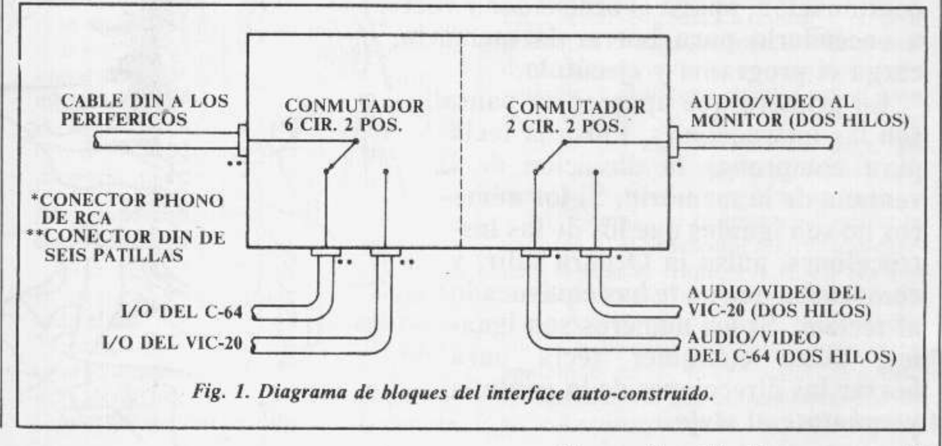


Fig. 1. Diagrama de bloques del interface auto-construido.

Viaje al centro de tu Commodore

Philip I. NELSON (RUN EEUU)

Traducido por Valerie SHANKS y adaptado por Alvaro IBAÑEZ

Vente de viaje con nosotros para explorar los misterios de la memoria de tu ordenador

¿Alguna vez has deseado poder ver lo que ocurre dentro de tu ordenador? El programa "Fisgon" convierte tu pantalla en una ventana móvil que puedes utilizar para emprender un viaje gráfico de exploración por la memoria entera de tu Commodore.

Podrás ver cómo se almacena y se organiza un programa en Basic, investigar el sistema operativo y de lenguaje Basic en la ROM del ordenador y viajar por muchos otros puntos de interés. Hablaremos principalmente del C-64, pero el dueño del VIC-20 también encontrará mucho que puede ser aplicado a su máquina, dado que los dos ordenadores se basan en el microprocesador 6502.

Para que el programa "Fisgon" funcione correctamente, tecléalo tal y como aparece en el listado, sin añadir ningún espacio ni quitar ningún REM. El programa contiene lenguaje máquina (ML), y por lo tanto es imprescindible salvar una copia antes de ejecutarlo, por si te has equivocado al teclearlo. A continuación, apaga el ordenador y vuelve a encenderlo para borrar la memoria, carga el programa y ejecútalo.

Lo primero que aparece en pantalla son las instrucciones. Pulsa la tecla A para comprobar la dirección de la ventana de la memoria. Si los números no son iguales que los de las instrucciones, pulsa la Q para salir, y comprueba que no te has equivocado al teclear. Si los números son iguales, pulsa cualquier tecla para borrar las direcciones de la pantalla y apúntate al viaje.

Si pulsas la tecla del cursor hacia abajo aparecen las direcciones de la parte inferior de la memoria, una línea de pantalla (40 bytes) a la vez. Deja pasar unas cuantas líneas a partir de la dirección 12288, donde empiezan las instrucciones; la pantalla empezará a llenarse de gráficos que no tienen ningún sentido.

Cada carácter gráfico en la pantalla representa una sola célula (dirección) de memoria en el ordenador. Existen 65.536 células en total, numeradas desde el cero hasta el 65.535 y cada célula siempre contiene algún número dentro del rango de cero hasta 255.

Puedes descubrir el número almacenado en cualquier dirección de pantalla simplemente consultando su carácter en la tabla de códigos de pantalla en el manual del usuario. Por ejemplo, el carácter * localizado a la izquierda superior de las instrucciones significa que la dirección 12288 contiene un valor de 42. La 12289 está en blanco, lo que quiere decir que el valor que contiene es de 32, etc. Ten en cuenta de que el valor cero está representado por el carácter @, y los caracteres de video inverso representan los valores de 128 hasta 255.

Ya que sabes esto, podrás deducir el valor almacenado en cualquier dirección y podrás identificar las direcciones utilizando tu comprobador de direcciones y unas sencillas operaciones matemáticas. Pero no pierdas el tiempo intentando interpretar esta "basura". Es simplemente "RAM libre", un espacio de memoria no utilizado que no contiene nada que tenga sentido. Esto aparece en cantidad antes de llegar al programa en Basic, asi que hay que seguir pasando líneas hasta que el comprobador de direcciones indique que has llegado a las direcciones 2084-3047. Pulsa la tecla del cursor si te pasas de largo.

¿Ya has encontrado el sitio? Si no encuentras las palabras en español pulsa las teclas Commodore y Shift simultáneamente. En la primera línea aparece la palabra Fisgón, con asteriscos al lado. Esta constituye la primera línea del programa en Basic, que contiene un REM. Hasta allí todo parece bastante familiar, y aparecen más palabras en español en otros sitios, pero ¿qué ha pasado con el resto del programa?

No te preocupes, las líneas que has tecleado siguen estando allí. Lo que pasa es que estás estudiando la versión "sólo para ordenadores". Para ahorrar tiempo y espacio, el ordenador abrevia las líneas en Basic a medida que las va almacenando en la RAM, comprimiendo cada palabra clave en Basic para que quede como un símbolo especial de un carácter llamado "token".

Compara esta versión abreviada con la línea 2 del listado, y verás que el comando Poke se abrevia como una W invertida. El "token" para GOSUB es una M invertida, etc.

Para que te des cuenta de la cantidad de espacio que se ahorra, piensa que este



bloque de 1.000 bytes contiene el programa "Fisgón" hasta casi el final de la línea 29. Si estas mismas líneas se presentaran en forma de listado, ocuparían alrededor de 1.650 bytes, casi las dos terceras partes más del espacio de memoria. Y esto es sólo parte del programa.

Si abrevias el texto del programa también ahorras espacio, dado que el ordenador es capaz de reconocer el "token" de un byte y seguir avanzando, en vez de tener que tragarse los tres o cuatro bytes de una palabra en Basic. El ordenador resulta más breve todavía al ser capaz de unir el final de una línea con el principio de otra. El ordenador marca el final de una línea para su propia referencia con un cero (busca el carácter @). También se abrevian los números de línea del programa, pero hay otras cosas que no pueden ser reducidas.

Como ya has visto, los REM (como en la línea 1) tienen que ser almacenados en la memoria de la misma forma que aparecen en el listado. De esta forma, puedes leer tus comentarios (REM) en las mismas "entrañas" del ordenador. Ocurre lo mismo con los nombres de las variables y los "arrays" en Basic y con cualquier carácter que metes entre comillas, como una sentencia print, por ejemplo. También quedan los espacios, con la excepción de los que aparecen después de un número de línea. Este almacenamiento literal de ciertas partes del texto normalmente no crea problemas para el potente C-64, pero el uso innecesario de los REM con el VIC-20 constituyer una pérdida de memoria que puede resultar cara.

Almacenamiento de variables

Esta zona de almacenamiento de texto es estática, y la podrías estar mirando todo el día, pero aquí no cambiaría nada. Tu programa en Basic tiene que ser almacenado como un texto fijo que el ordenador leerá mientras lo está ejecutando Como un estudiante diligente, comienza desde el principio y va por orden, ejecutando las instrucciones a medida que avanza.

El almacenamiento de un texto en Basic en el C-64 normalmente empieza en la dirección 2048, de modo que el ordenador no tiene ningún problema en localizar el comienzo. En realidad, la dirección 2048 siempre tiene que contener un cero, de modo que el texto (incluyendo los números de línea) realmente empieza en la dirección 2049. Pero algunos programas son más largos que otros, y el ordenador necesita saber cuándo ha llegado al final de un texto. Por eso el ordenador coloca un indicador. Cuando almacena un programa en Basic, el ordenador marca el punto en la memoria donde finaliza el texto mediante el almacenamiento de dos ceros adicionales justo después del cero que marca el final de la última línea del programa en Basic.

Ahora vuelve al principio de la memoria para ver esto. Para que fuese más fácil de encontrar, dejé unas líneas en blanco donde sabía que el programa "Fisgón" terminaría en la RAM, y puse un comentario (REM) en la última línea que dice "—Fin de programa en Basic—Principio variables—". Ten en cuenta que los espacios oscuros en blanco normalmente no están presentes. Yo los puse allí para que esta primera búsqueda resultara más fácil.

¿Has encontrado el último REM? El indicador de tres ceros aparece después de la flecha. Si avanzas esta línea hasta el principio de la pantalla y haces uso del comprobador de direcciones, puedes calcular fácilmente que el último carácter del texto en Basic ocupa la dirección 5020, seguido por el indicador de fin de texto. Pero si esto es el final, ¿quién añadió todo lo demás a tu programa, y qué significa?

El culpable es el ordenador, que necesita más RAM para ejecutar tu programa que la que ocupa el texto solo. Antes de poder realizar los comandos, el ordenador tiene que reservar tres zonas adicionales de memoria para su propio uso espacio para el almacenamiento de variables, "arrays" y "strings".

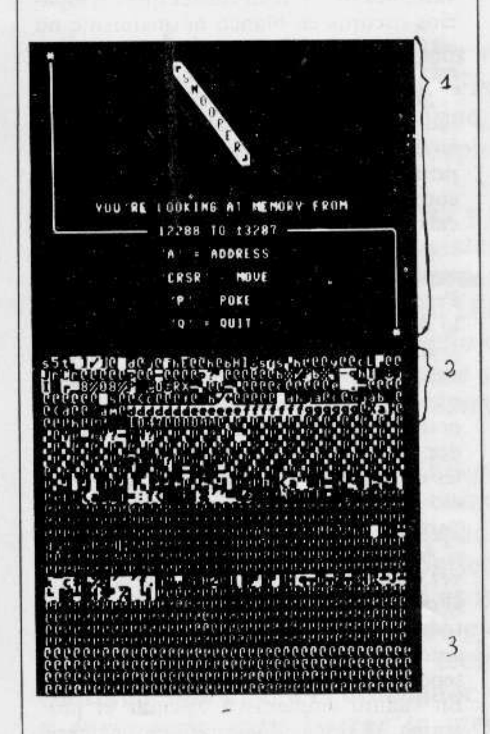
A lo mejor te has preguntado alguna vez, al hacer un programa en Basic, cómo el ordenador se acuerda de los valores de todas las variables que estás utilizando. El sistema que utiliza es sorprendentemente sencillo, y lo estás viendo ahora mismo. En cuanto empiezas a ejecutar el programa, el ordenador establece una serie de casillas en la memoria, una para cada variable, y almacena el valor actual para cada variable en su propio sitio. Este espacio de almacenamiento para las variables siempre empieza después (encima) del texto del programa en Basic.

Por ejemplo, si tu programa utiliza la variable A, el ordenador reserva una casilla y le pone la etiqueta A. Cada vez que cambias el valor de A con una sentencia como A=A+1, por ejemplo, el ordenador vuelve a la casilla A para convertir su contenido en el nuevo valor. De esta forma, cada vez que utilizas la variable A en una sentencia como C=A*B, el orde- ▶

nador volverá a la casilla A para saber el valor actual de A antes de utilizarla en el cálculo.

El ordenador empieza a fijar esta serie de casillas para las variables nada mas recibir el orden de ejecución, y crea un espacio de almacenamiento para cada variable en el mismo orden que las encuentra durante la ejecución del programa. Para que veas cómo funciona en el programa "Fisgón", mira el listado donde aparece un GOSUB de la línea 2 a la línea 39. De esta forma se fija la rutina de lenguaje máquina y la representación de las instrucciones en la memoria.

Por lo tanto, las primeras variables que encuentra el ordenador es la Z en la línea 2, seguida por la J y la Q en la línea 41 y la K en la línea 86. Fíjate bien en las células de memoria después del marcador de fin de texto, y encontrarás cada uno de estos espacios para variables etiquetado con su propia letra. Estas variables no están en uso, de modo que sus casillas están inactivas. Pero las dos siguientes casillas demostrarán cómo funciona el almacenamiento de las variables.



1. Esto aparece nada más teclear RUN. La imagen en pantalla se desplazará hacia la derecha pero vuelve en seguida al centro. Aquí empieza el viaje por la memoria del ordenador.

2. Estas son las posiciones de 0 a 255, es decir, la "página cero". Esta zona resulta muy útil para los programadores en lenguaje máquina.

3. El sistema operativo de la memoria también tiene que disponer de una zona de trabajo.



El programa "Fisgón" utiliza dos variables, LO y HI, para saber dónde se encuentra en cada momento. ¿Has encontrado LO y HI? Ahora avanza la memoria en pantalla hacia arriba o hacia abajo, línea por línea. A medida que va pasando, aparecen nuevos caracteres justo al lado de las etiquetas de las variables, dado que el ordenador almacena los nuevos valores de LO y HI en sus casillas respectivas.

Detrás de LO y HI se encuentra un espacio para la única variable de "strings", A$. Esta casilla es la más activa. Si no la has encontrado todavía, manten pulsada la barra de espacio, y observa mientras el ordenador introduce los nuevos valores a toda velocidad en la casilla de la variable A$. Ocurre así porque la línea 4 del programa contiene la sentencia GET A$, y pasa una y otra vez por dicha línea.

Cada vez que pasa, el ordenador almacena un nuevo valor para A$ si se pulsa una tecla distinta. (Como verás más adelante, otra cosa está ocurriendo en la parte superior de la memoria). Las siguientes dos variables, A y B, no se actualizan hasta que utilizas el comprobador de direcciones. Verás que los valores se modifican dentro de sus casillas si tu compruebas tu dirección cada vez que se avanza una línea hacia arriba o hacia abajo.

Cuando el programa llega por primera vez a la rutina Poke, el ordenador reservará un espacio para las últimas dos variables, C y D. Resulta divertido observar como los caracteres se modifican dentro de estas casillas, pero no te molestes en intentar descifrar su significado, a no ser que entiendas sistemas de numeración tan complicados como "Código EBCDC". Acepta de buena fe que tu ordenador sabe lo que está haciendo.

Dado que cada programa Basic termina en un sitio distinto, y el espacio para el almacenamiento de variables se agrega al final del texto en Basic, la dirección donde comienza el almacenamiento de variables dependerá de la longitud de tu programa. El tamaño de la zona de almacenamiento de variables dependerá del número de variables que utilizas.

Si necesitas ahorrar espacio, resulta más económico utilizar las mismas variables "corrientes" una y otra vez en una serie de bucles For... Next, como en las líneas 41 a 87 del programa "Fisgón". De no ser así, se reservará memoria para una parte de variables que sólo realizan una tarea al principio del programa y que no hacen nada más el resto del tiempo.

Almacenamiento de "arrays"

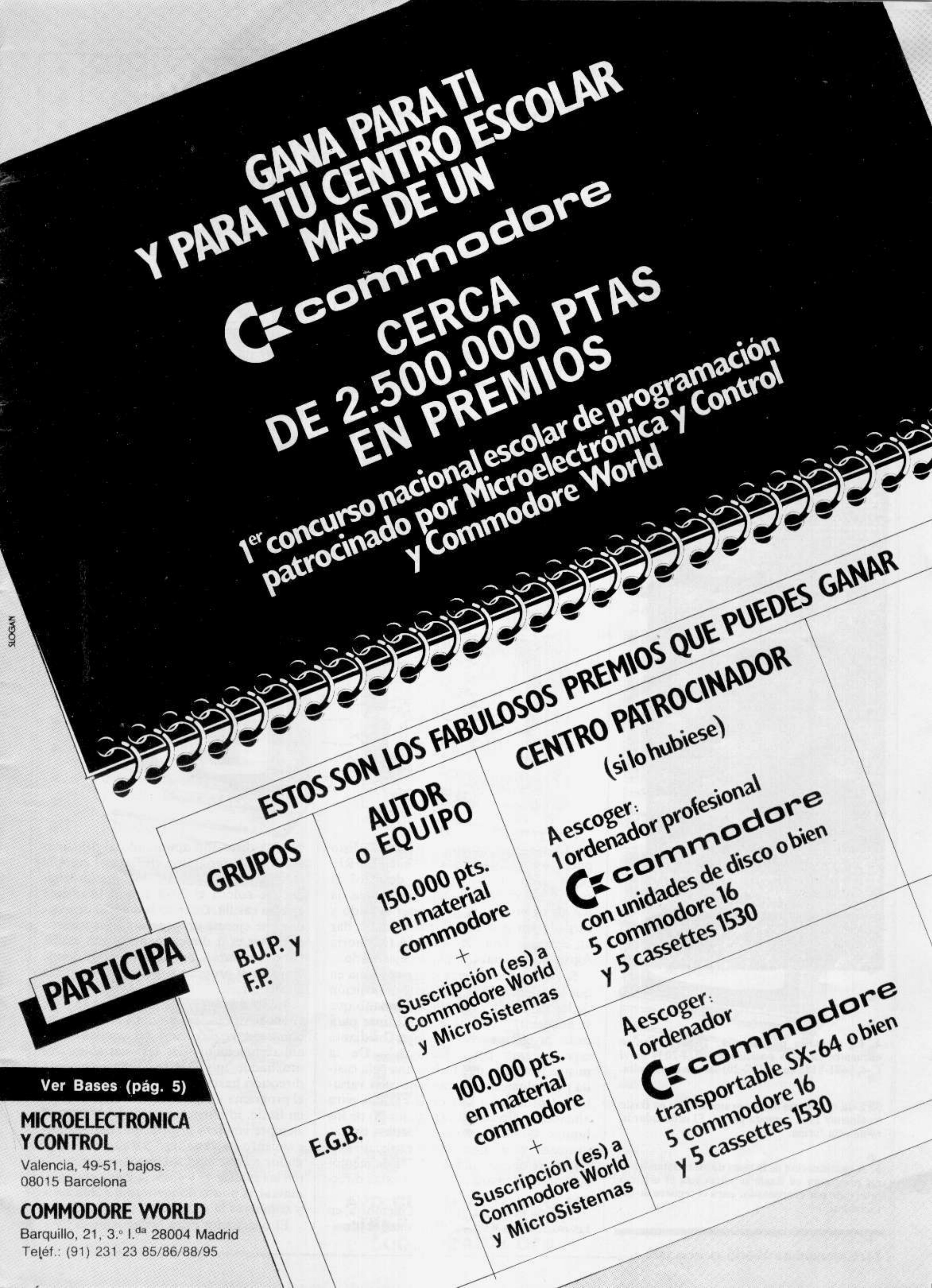
Justo encima de la zona para el almacenamiento de variables, se encuentra otra zona especial, el espacio para el almacenamiento de "arrays". Un "array" funciona como una tabla o lista de datos dentro de tu programa. En primer lugar introduces todos los valores, y los sacas siempre que los necesitas, simplemente haciendo referencia a su posición en el "array".

El programa "Fisgón" no necesita hacer esto, pero para demostrar cómo se hace, incluye un "array" que no hace nada y que se llama ARRAY. Cada vez que pulsas la tecla Z, el ordenador asignará un nuevo valor (al azar) al siguiente espacio (elemento) en la tabla ARRAY. Para ver cómo ocurre esto, tienes que encontrar la cabecera (header) que pone la etiqueta y define el "array". Primero encontrarás los dos primeros caracteres del propio nombre, AR. A continuación siguen las cinco posiciones que definen (en formato comprimido) el tamaño y el tipo del "array".

Este "array" es de "coma flotante", de modo que el ordenador ha reservado cinco células para almacenar el valor para cada elemento. El espacio se guarda llenando las células de ceros. A medida que vayas definiendo cada elemento pulsando la tecla Z, el ordenador va metiendo los nuevos valores en estos espacios. Los valores son todos números enteros, que pueden ser almacenados en dos células, de modo que tres de las cinco células en cada espacio quedan sin usar.

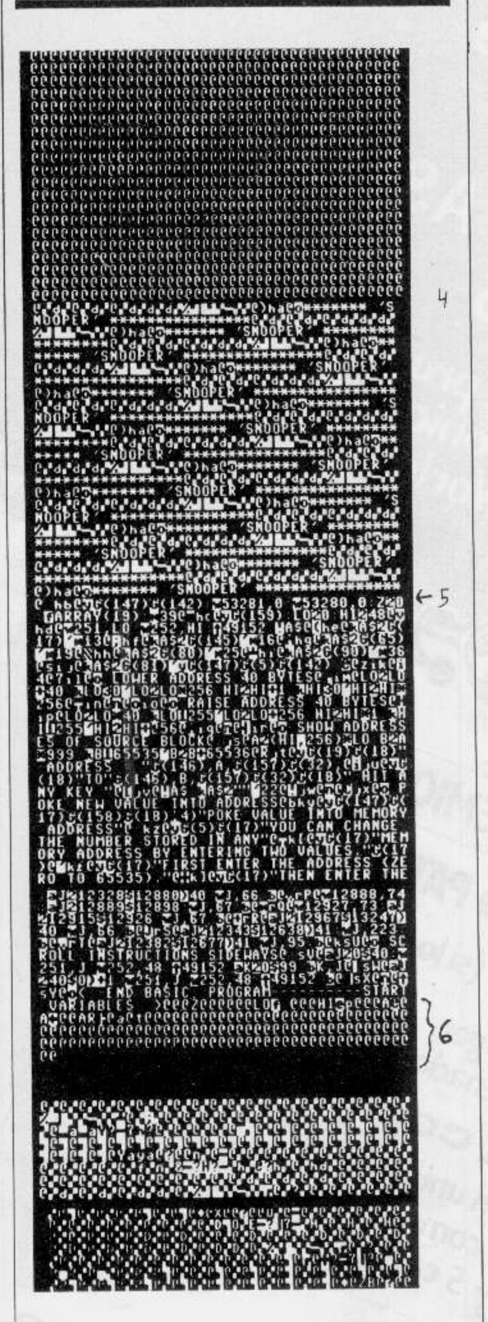
Pulsa la tecla Z unas cuantas veces, para ver cómo funciona. Recuerda que el "array" tiene 20 elementos, aunque ha sido dimensionado con la sentencia DIMARRAY (19). En este caso, como en muchos otros, el ordenador cuenta cero como un número, de modo que el primer elemento es ARRAY (0), seguido por ARRAY (1), etc.

El punto que indica el fin de las variables puede modificarse a mitad de un programa si el ordenador encuentra una nueva variable por primera vez. Cuando esto ocurre, toda la zona de almacenamiento de "arrays" volverá a ser "situada" en la memoria para hacer hueco para almacenar la nueva variable. El ARRAY de "Fisgón" no ocupa mucho sitio en la RAM, pero ten en cuenta que el espacio para los "arrays", igual que el espacio para las variables, no se da gratuitamente.



GANA PARA TI
Y PARA TU CENTRO ESCOLAR
MAS DE UN
commodore
CERCA
DE 2.500.000 PTAS
EN PREMIOS
1er concurso nacional escolar de programación
patrocinado por Microelectrónica y Control
y Commodore World
ESTOS SON LOS FABULOSOS PREMIOS QUE PUEDES GANAR
GRUPOS
AUTOR o EQUIPO
CENTRO PATROCINADOR (si lo hubiese)
B.U.P. y F.P.
150.000 pts. en material commodore + Suscripción (es) a Commodore World y MicroSistemas
A escoger: 1 ordenador profesional commodore con unidades de disco o bien 5 commodore 16 y 5 cassettes 1530
E.G.B.
100.000 pts. en material commodore + Suscripción (es) a Commodore World y MicroSistemas
A escoger: 1 ordenador commodore transportable SX-64 o bien 5 commodore 16 y 5 cassettes 1530
PARTICIPA
Ver Bases (pág. 5)
MICROELECTRONICA Y CONTROL
Valencia, 49-51, bajos.
08015 Barcelona
COMMODORE WORLD
Barquillo, 21, 3.º I.da 28004 Madrid
Teléf.: (91) 231 23 85/86/88/95
SLOGAN

Tu ordenador no puede ejecutar un programa en Basic sin haber reservado antes la memoria necesaria para estas zonas adicionales, lo que significa que el programa ocupa más memoria después de





ejecutarse que cuando se carga. Esto puede ser verificado con la función FRE. ¿Qué es lo que se aprende de esto? Si solamente el texto ocupa casi toda la RAM, tu programa podría ser salvado y cargado sin problemas, pero podría dar un error de Out Of Memory (Memoria Agotada) en cuanto intentes ejecutarlo.

Seguramente querrás saber la forma en que el ordenador se entera de la posición de las variables y los "arrays", dado que se encuentran en una zona distinta para cada programa. No lo hace mediante espejos, sino con "punteros". De la misma forma en que utiliza una sola casilla para almacenar el valor de cada variable, el ordenador usa otras casillas para almacenar direcciones (posiciones) de los límites de estas zonas. Dadas ciertas "rarezas" en el sistema (que explicaré más tarde), se necesitan dos células de memoria conjuntas para almacenar estas direcciones.

En una parte inferior de la memoria, en las posiciones 43 y 44, el ordenador almacena la dirección apuntando al principio el texto del programa en Basic. Para que resulte más fácil, al encenderse el ordenador, se coloca el 2049 automáticamente en esta casilla. Cuando le dices al ordenador que ejecute un programa, lo primero que hace es ir directamente a esta casilla para enterarse de la posición donde empieza el programa, y luego lo empieza a ejecutar.

Justo al lado, en las posiciones 45 y 46, se encuentra la casilla donde el ordenador almacena la dirección del comienzo del almacenamiento de las variables. El ordenador no puede almacenar esta dirección hasta que no termine de cargar el programa y sepa dónde termina el texto en Basic. El almacenamiento de "arrays" siempre empieza al principio del almacenamiento de variables, para que el ordenador pueda usar solamente un puntero (en las células 47 y 48 de la memoria) para marcar el punto donde termina una zona y comienza la siguiente.

El ordenador mira las posiciones 49 y

4. La memoria de pantalla. Todo lo que se encuentre en estas posiciones (1024-2039 en el C-4, 7681-8185 en el VIC-20) sale en la pantalla.

5. Esta es la parte del programa escrita en Basic (realmente es la segunda parte). El ordenador lo ve de esta forma.

6. A continuación de la zona de almacenamiento del programa en Basic se encuentra el espacio reservado por el ordenador para el registro de las variables.

50 para saber dónde termina el almacenamiento de los "arrays". No es necesario que comprendas estos punteros para poder realizar una programación simple



en Basic, ya que el ordenador lo hace él solo. Pero podrías disponer de unos cuantos trucos si sabes cómo funcionan.

Hasta la página cero

De momento, vamos a olvidarnos del texto en Basic y seguir viajando hacia la parte inferior de la memoria. Justo debajo de tu texto se encuentran ocho posiciones (2040-2047) para los punteros de los gráficos sprite. Aquí no hay nada interesante, dado que m no utiliza sprites. Sin embargo, la zona justo debajo de la 2024 se está usando mucho, y si te fijas bien en esta zona verás que la pantalla empieza a hacer unas cosas muy raras.

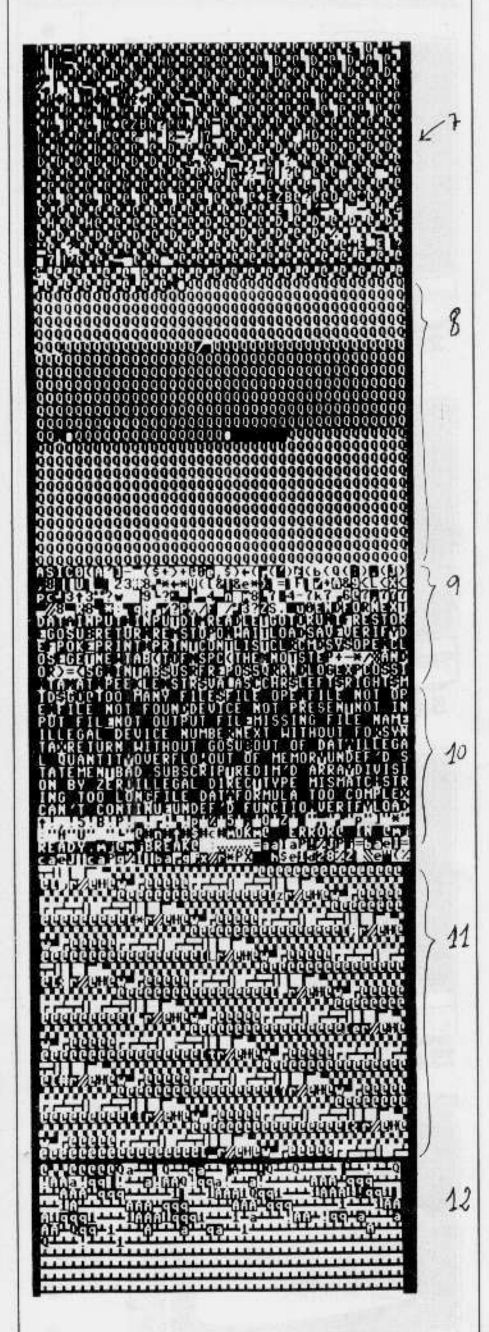
Lo que has encontrado ha sido el bloque de memoria de 1.000 bytes reservado para la memoria de pantalla (1024-2023). El ordenador utiliza esta zona para representar mensajes, gráficos, o lo que sea en la pantalla de tu televisión. Bajo circunstancias normales, si introduces cualquier valor de cero hasta 255 en cualquiera de estas 1.000 posiciones mediante un poke, aparecerá un carácter en pantalla.

Pero no intentes interpretar los caracteres que ves flotando por la memoria de pantalla ahora —no tienen sentido—. "Fisgón" está copiando y volviendo a copiar las zonas más altas de la memoria de pantalla en las zonas más bajas a medida que tú vayas cambiando las direcciones. Juega un poco si quieres. Verás que si cambias las líneas una por una, se crean unos efectos muy diferentes de los que ves cuando esta zona se va pasando normalmente. La pantalla no volverá a comportarse de una forma normal hasta que no hayas pasado completamente por debajo de la memoria de pantalla.

Te evitará problemas si no te olvidas nunca que la memoria de pantalla y el texto en Basic viven puerta con puerta en el C-64. Si escribes un juego de gráficos donde tienes que introducir los caracteres en la pantalla mediante unos pokes, pero te descuidas y los caracteres de tu juego pasan más allá de la posición 2023 y entran en las posiciones por encima de la 2047, harás uns pokes en el texto en Basic y tus instrucciones serán sustituidas por "basura". Para que esto no te ocurra, tienes que tener cuidado de que no te salga ningún poke más allá de las posiciones 1024-2023, o tomar otras medidas para proteger tu texto.

Ahora estás muy cerca del final, y tu destino es la famosa "página cero". Vamos a pasar volando por las posiciones 256-1022, donde sólo se encuentra el espacio de trabajo y cosas raras para el propio ordenador. Desplázate hacia abajo hasta que el comprobador de direcciones indique que te encuentras en las posiciones 8-1007. No te preocupes si por descuido te desplazas "por debajo" de la posición cero. El programa envolverá toda la memoria del ordenador y añadirá unos de sus bytes más altos a los más bajos.

La página cero, es decir, las 256 posiciones de memoria más bajas del ordenador, constituye una de las zonas más importantes y útiles. Resulta tan especial porque la dirección de cada posición es tan pequeña que cabe dentro de una sola célula de memoria, mientras que toda las



7. Una zona a veces vacía, a veces llena de "basura" o de RAM, una especie de tierra de nadie.

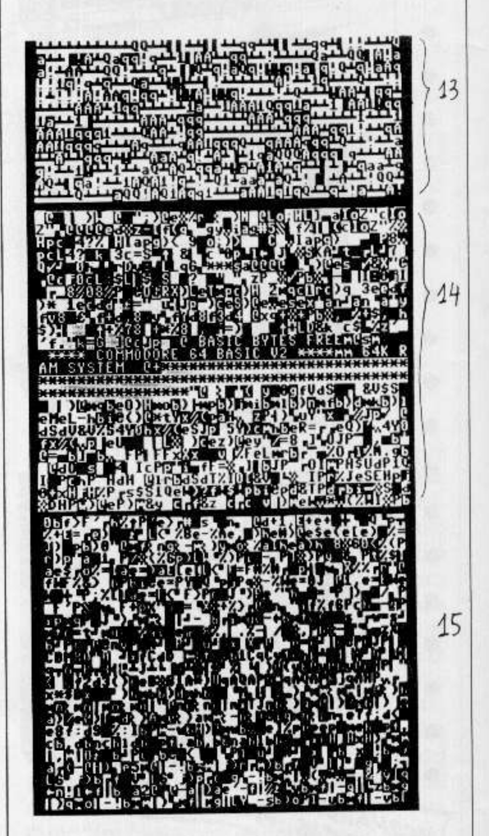
8. Zona de almacenamiento de las variables de "strings". La letra Q indica que un cursor hacia arriba o hacia abajo ha sido añadido al "string" A$.

9. Tabla de comandos en Basic.

10. Aquí se encuentran los mensajes de error.

11. Area de control del Interface de Video VIC-II.

12. Sonido (registros del SID).



13. RAM de color.

14. Mensajes de encendido de Commodore.

15. Una imagen parcial del kernal. (Tiene una longitud de 8K).

demás direcciones ocupan dos células (hablaremos de esto más adelante).

Por este motivo, esta zona resulta ser tan importante para los programadores en lenguaje máquina —de hecho resulta tan valiosa que los fabricantes del ordenador han incluido en esta página unas zonas de almacenamiento especiales y muchas otras cosas que el ordenador necesita para sus propias operaciones—. A diferencia de la zona permanente de almacenamiento de texto, siempre hay actividad aquí.

Sin embargo, esta zona es RAM, de modo que el ordenador puede usar unos pokes para introducir los nuevos valores en cualquier célula de la página cero. Para los curiosos y los que disfrutan con los experimentos, existen cuatro células (251-254) reservadas para que puedas hacer lo que quieras con ellas. "Fisgón" utiliza dos de ellas (251-252) para almacenar un puntero de "dirección original" para la subrutina en lenguaje máquina.



Si quieres ver tu propia casilla, busca primero la línea de pantalla que empieza en la posición 248. Si haces un "scroll" hacia arriba o hacia abajo verás que cambia el carácter dentro de la posición 251. No confundas este espacio de almacenamiento con la zona más grande de 256-266 donde aparece un número decimal —esto es una pequeña zona de trabajo utilizada por el ordenador—. Todas las direcciones aquí abajo caben en una célula, de modo que la segunda mitad de nuestra casilla, 252, ahora contiene solamente un cero.

En el próximo artículo, iremos de excursión por debajo de la página cero, hasta la parte superior de la memoria. Podrás ver el sistema operativo ROM, las zonas para el color, el sonido y el manejo de periféricos, y finalmente terminaremos estudiando un poco los pokes, peeks y los cálculos del direccionamiento.

```
1 REM* 'VIC FISGON' *
2 PRINTCHR$(147)CHR$(14):POKE36879,8
:Z=0:DIMARRAY(19):GOSUB39
3 LO=0:HI=16
4 POKE251,LO:POKE252,HI:SYS828:GETA$
5 IFA$=CHR$(17)THENGOSUB13
6 IFA$=CHR$(145)THENGOSUB16
7 IFA$=CHR$(65)THENGOSUB19
8 IFA$=CHR$(80)THENGOSUB25
9 IFA$=CHR$(90)THENGOSUB36
10 IFA$=CHR$(81)THENPRINTCHR$(147)CH
R$(142):END
11 GOTO4
12 REM BAJA DIRECCION
13 LO=LO-22:IFLO<0THENLO=LO+256:HI=H
I-1:IFHI<0THENHI=HI+256
14 RETURN
15 REM SUBE DIRECCION
16 LO=LO+22:IFLO>255THENLO=LO-256:HI
=HI+1:IFHI>255THENHI=HI-256
17 RETURN
18 REM MUESTRA ADDRESS
19 A=(HI*256)+LO:B=A+505:IFB>65535TH
ENB=B-65536
20 PRINT"[HOM][GRN][RVSON]ADDRESS"A"
[CRSRL][SPC]TO"B"[WHT]"
21 GETA$:IFA$=""THEN21
22 PRINTCHR$(19)"[22SPC]"
23 RETURN
24 REM HACE UN POKE
25 PRINT"[CLR][YEL][RVSON][SPC]POKE[SPC]
UN[SPC]VALOR":PRINT"[CRSRD][WHT][SHIFTC]
AMBIA[SPC]EL[SPC]CONTENIDO[SPC]DE";:
PRINT"[CRSRD]UNA[SPC]DIRECCION"
26 PRINT"[CRSRD]ENTRANDO[SPC]DOS[SPC]
NUMEROS.":PRINT"[CRSRD][SHIFTE]L[SPC]
PRIMERO[SPC]LA[SPC]POSICION";:PRINT"
[CRSRD]Y[SPC]EL[SPC]SEGUNDO[SPC]EL"
27 PRINT"[CRSRD]VALOR[SPC]QUE[SPC]QU
IERES":PRINT"[CRSRD]COLOCAR."
28 INPUT"[YEL][CRSRD][SHIFTE]NTRA[SPC]
2[SPC]VALORES";C,D:POKEC,D:PRINT"[WHT]
ALGUNO[SPC]MAS?[SPC]'S'=SI"
29 GETA$:IFA$=""THEN29
30 IFA$="S"THEN28
31 PRINTCHR$(147):RETURN
35 REM ARRAY MARCADOR
36 ARRAY(Z)=INT(RND(1)*255):Z=Z+1:IF
Z=20THENZ=0
37 RETURN
38 REM RUTINA C.M.
39 IFPEEK(828)=160ANDPEEK(850)=96THE
NRETURN
40 FORJ=828TO850:READQ:POKEJ,Q:NEXT
41 DATA 160,0,177,251,153,0,30,200,2
08,248
42 DATA 230,252,160,0,177,251,153,0,
31,200
43 DATA 208,248,96
44 FORJ=5482TO5635:POKEJ,32:NEXT
45 REM MOVIMIENTO LATERAL
46 FORJ=0TO30:POKE251,J:POKE252,16:S
YS828:FORK=1TO10:NEXTK,J
48 RETURN
49 REM <-FIN PROGRAMA EN BASIC--INIC
IO DE VARIABLES>
```

```
1 REM****** 'FISGON' ***********
*
2 PRINTCHR$(147)CHR$(142):POKE53281,
0:POKE53280,0:Z=0:DIMARRAY(19):GOSUB
39
3 PRINTCHR$(159):LO=0:HI=48
4 POKE251,LO:POKE252,HI:SYS49152:GET
A$
5 IFA$=CHR$(17)THENGOSUB13
6 IFA$=CHR$(145)THENGOSUB16
7 IFA$=CHR$(65)THENGOSUB19
8 IFA$=CHR$(80)THENGOSUB25
9 IFA$=CHR$(90)THENGOSUB36
10 IFA$=CHR$(70)THENPRINTCHR$(147)CH
R$(142):END
11 GOTO4
12 REM BAJA 40 BYTES EL FINAL DE LA
MEMORIA
13 LO=LO-40:IFLO<0THENLO=LO+256:HI=H
I-1:IFHI<0THENHI=HI+256
14 RETURN
15 REM SUBE 40 BYTES EL FINAL DE LA
MEMORIA
16 LO=LO+40:IFLO>255THENLO=LO-256:HI
=HI+1:IFHI>255THENHI=HI-256
```

```
17 RETURN
18 REM MUESTRA LA DIRECCION DEL BLOQ
UE FUENTE
19 A=(HI*256)+LO:B=A+999:IFB>65535TH
ENB=B-65536
20 PRINTCHR$(19)CHR$(18)"DIRECCIONES
"CHR$(146);A;CHR$(157)CHR$(32);
21 PRINTCHR$(18)"[CRSRL]"CHR$(146);-
B;CHR$(157)CHR$(32)CHR$(18)"[SPC]PUL
SA[SPC]UNA[SPC]TECLA"
22 GETA$:IFA$=""THEN22
23 RETURN
24 REM POKE EN UNA DIRECCION
25 PRINTCHR$(147)CHR$(17)CHR$(158)CH
R$(18)TAB(4)"POKE[SPC]EN[SPC]LA[SPC]
MEMORIA"
26 PRINTCHR$(5)CHR$(17)"PUEDES[SPC]C
AMBIAR[SPC]EL[SPC]CONTENIDO[SPC]DE[SPC]
CUALQUIER";
27 PRINTCHR$(17)"POSICION[SPC]DE[SPC]
MEMORIA[SPC]INTRODUCIENDO[SPC]DOS"
28 PRINTCHR$(17)"VALORES.":PRINT"[CRSRD]
[SPC]PRIMERO[SPC]ENTRA[SPC]LA[SPC]DI
RECCION[SPC](0[SPC]A[SPC]65535)"
30 PRINTCHR$(17)"QUE[SPC]QUIERES[SPC]
CAMBIAR."CHR$(17)CHR$(17)CHR$(158)
31 INPUT"ENTRA[SPC]2[SPC]VALORES";C,
D:POKEC,D:PRINTCHR$(17)CHR$(5)"OTRO[SPC]
MAS[2SPC]'S'=SI[SPC]"
32 GETA$:IFA$=""THEN32
33 IFA$="S"THENPRINTCHR$(158):GOTO31

34 PRINTCHR$(159):RETURN
35 REM  DEFINE ELEMENTO DEL ARRAY
36 ARRAY(Z)=INT(RND(1)*255):Z=Z+1:IF
Z=20THENZ=0
37 RETURN
38 REM CARGADOR BASIC PARA LA RUTINA
 DE CODIGO MAQUINA
39 IFPEEK(49152)=160THENRETURN
40 POKE214,12:PRINT:PRINTTAB(10)CHR$
(18)CHR$(5)"[2SPC]UN[SPC]MOMENTO[2SPC]
"
41 FORJ=49152TO49200:READQ:POKEJ,Q:N
EXTJ
42 DATA 160,0,177,251,153,0,4
43 DATA 200,208,248,230,252
44 DATA 160,0,177,251,153,0,5
45 DATA 200,208,248,230,252
46 DATA 160,0,177,251,153,0,6
47 DATA 200,208,248,230,252
48 DATA 160,0,177,251,153,0,7
49 DATA 200,192,232,208,246,234
50 REM  LA SIGUIENTE RUTINA DE C.M.
LIMPIA LA MEMORIA DE COLOR
51 FORJ=49201TO49220:READQ:POKEJ,Q:N
EXT
52 DATA 162,0,169,1,157,0,216
53 DATA 157,0,217,157,0,218,157
54 DATA 0,219,232,208,241,96
55 REM DATOS DE INSTRUCCIONES EN LA

PANTALLA
56 DATA 32,160,134,137,147,135
57 DATA 143,142,160,5,19,20,1
58 DATA 19,32,22,9,5,14,4,15
59 DATA 32,12,1,32,13,5,13,15
60 DATA 18,9,1,32,4,5,19,4
61 DATA 5,49,50,50,56,56,32,32
62 DATA 1,32,49,51,50,56,55,39
63 DATA 1,39,32,61,32,1,4,4,18
64 DATA 5,19,19,32,40,4,9,18,5,3,3,9
,15,14,41
65 DATA 39,3,18,19,18
66 DATA 39,32,61,32,13,15,22,5,18
67 DATA 39,16,39,32,61,32,16
68 DATA 15,11,5,39,6,39,32
69 DATA 61,32,6,9,14,32
70 REM ESCRIBE LAS INSTRUCCIONES
71 FORJ=12288TO13327:POKEJ,32:NEXT:F
ORJ=5128TO5327:POKEJ,32:NEXT
72 FORJ=12342TO12678STEP41:READQ:POK
EJ,Q:NEXT
73 FORJ=12813TO12841:READQ:POKEJ,Q:N
EXT
74 FORJ=12900TO12913:READQ:POKEJ,Q:N
EXT
75 FORJ=12980TO13004:READQ:POKEJ,Q:N
EXT
76 FORJ=13060TO13073:READQ:POKEJ,Q:N
EXT
77 FORJ=13141TO13150:READQ:POKEJ,Q:N
EXT
78 FORJ=13221TO13230:READQ:POKEJ,Q:N
EXT
79 POKE12288,42:POKE13287,42:FORJ=12
328TO12880STEP40:POKEJ,66:NEXT
80 POKE12888,74:FORJ=12889TO12898:PO
KEJ,67:NEXT
81 POKE12927,73:FORJ=12915TO12926:PO
KEJ,67:NEXT
82 FORJ=12967TO13247STEP40:POKEJ,66:
NEXT
83 FORJ=12343TO12638STEP41:POKEJ,223
:NEXT
84 FORJ=12382TO12677STEP41:POKEJ,95:
NEXT
85 REM SCROLL LATERAL DE INSTRUCCION
ES
86 FORJ=0TO40:POKE251,J:POKE252,48:S
YS49152:NEXTJ
87 FORJ=40TO0STEP-1:POKE251,J:POKE25
2,48:SYS49152:NEXT
88 RETURN
89 REM<--FIN DE PROGRAMA BASIC----PR
INCIPIO VARIABLES>
```

casa de software s.a.

PARA COMMODORE 64

PRACTICALC 64



MANUAL EN CASTELLANO

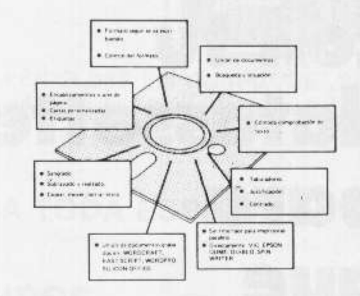
PROCESADOR DE TEXTO

35.000 caracteres, 240 columnas
Versión diskette: 21.500,–
Versión cartucho: 24.900,–
(grabación de documentos en diskette y cassette)

HOJA DE CALCULO

2.000 coordenadas
(funciones matemáticas, sort, gráficos...)
Versión cassette C-64: 15.500,–
Versión diskette C-64: 17.500,–
Versión cassette VIC-20: 11.500,–
Versión diskette VIC-20: 13.500,–

VIZAWRITE 64



MANUAL EN CASTELLANO

CONTABILIDAD 64 PROFESIONAL



VERSION: A	VERSION: B
300 cuentas	600 cuentas
3.000 apuntes	2.300 apuntes

- Listado de diario
- Balance de situación y de Sumas y Saldos
- Estractos de cuenta
- Cuenta de explotación, diario de cierre
- Mantenimiento de ficheros, utilitario...

EQUIPO NECESARIO:

- COMMODORE 64 y unidad de disco
- IMPRESORA MPS 801 (versión A y B)
- IMPRESORAS CENTRONICS (versión B)

P.V.P.: VERSION A: 24.550,–
VERSION B: 29.500,–

ADAPTADO AL PLAN GENERAL CONTABLE ESPAÑOL

PROGRAMAS EN CARTUCHO + DISKETTE

VIDEOCLUBS 64

1.800 videofilms
650 clientes por disco
Saldo de cada cliente
Altas y bajas videofilms
Búsqueda videofilms/clientes.

P.V.P.: 35.000,–

PROGRAMAS EN CASSETTE

CONTABILIDAD PERSONAL	6.000,–
CONTABILIDAD DOMESTICA	3.500,–
CONTABILIDAD DOMESTICA (VIC-20)	2.500,–
PERT	4.000,–
CALCULOS FINANCIEROS	3.000,–
CALCULOS ESTADISTICOS	1.900,–
FICHERO MONEDAS	3.000,–
FICHERO SELLOS	3.000,–
FICHERO BIBLIOTECA	3.000,–
FICHERO AGENDA	3.000,–
QUINIELAS	5.000,–

ADMINISTRACION DE FINCAS

30 inmuebles
400 inquilinos
Listado recibos
Propiedad vertical
Comunidades de propietarios

P.V.P.: 35.000,–

LAPIZ OPTICO Y SOFTWARE GRAFICO

P.V.P.: 12.500,–

JOY STICK DE PRECISION

P.V.P. 2.950,–

DE VENTA EN TIENDAS Y DISTRIBUIDORES ESPECIALIZADOS

casa de software, s.a.
c/. aragón, 272, 8.°, 6.ª
tel. 215 69 52
08007 barcelona

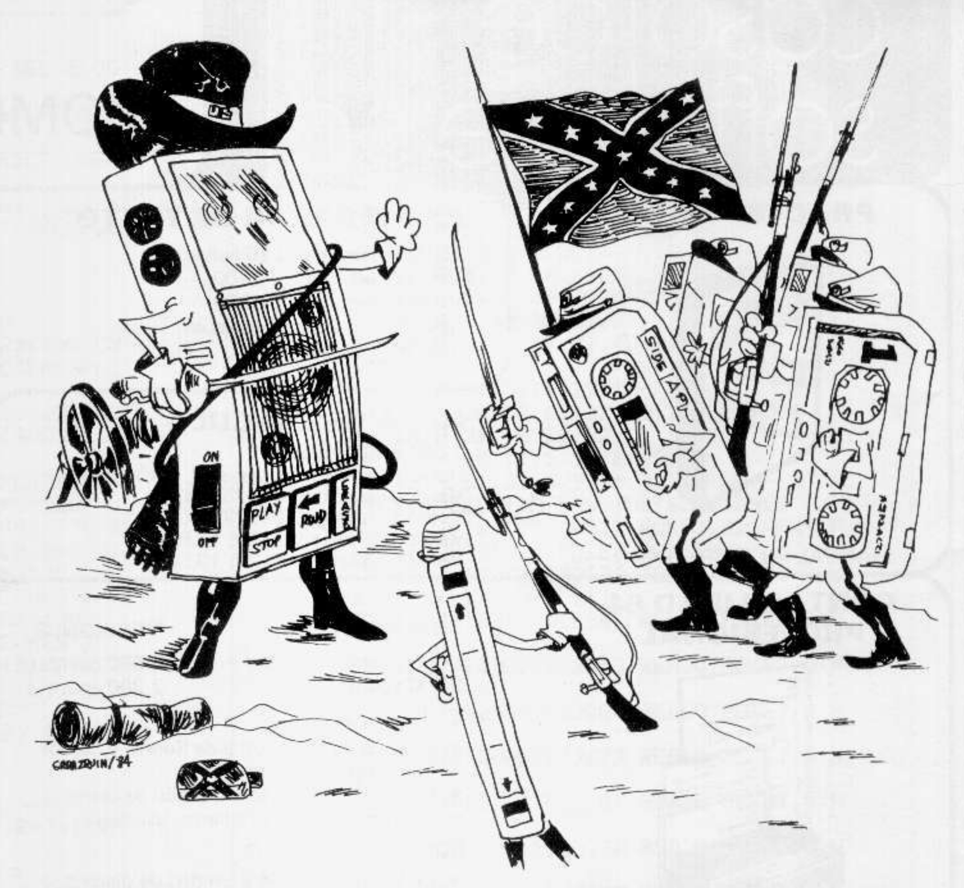
DESEO RECIBIR INFORMACION:

Nombre y dirección:

COMODORE 64 ES UNA MARCA REG. DE COMMODORE BUSSINES MACHINES, INC.

Por Diego ROMERO

Qué hacer con tu datassette para que entren esas cintas rebeldes



En alguna ocasión habréis tenido la oportunidad de intercambiar programas con alguien, pero al llegar a casa con la cinta grabada en otro datassette, ¡qué desilusión!, no hay manera de hacer entrar la cinta, unas veces da "LOAD ERROR", otras se para antes de cargar el programa, o sólo una parte y se queda "colgao en las alturas"... por cierto al datassette no le iría nada mal... un repaso a sus ajustes.

Vamos a ver cuáles son las posibles causas de que las cintas no carguen en nuestros ordenadores.

Uno de los principales problemas que suelen aparecer es que la cinta se frene; la solución para eso suele ser como muchos saben, darle unos golpecitos contra la mesa, ¡pero sin pasarse!, a la cinta para colocar todas las vueltas al mismo nivel, y darle varias pasadas a la cinta de un lado a otro con las teclas REWIND y FAST FORWARD, de este modo conseguiremos igualar la tensión y la posición de todas las vueltas de la cinta.

El otro problema suele ser la suciedad acumulada en la cabeza, su remedio es utilizar una de las cintas limpiadoras que están comercializadas para los equipos de alta fidelidad, ya que aunque el datassette tenga salida digital, su funcionamiento es similar al de los normales de sonido. También se podrían limpiar las cabezas de borrado y grabación/lectura con un algodón empapado en alcohol diluido, teniendo cuidado de no dejar restos de algodón enganchados en los mecanismos.

Si todo lo anterior falla, pasamos a lo más delicado por llamarlo de alguna manera, se trata del ajuste de la posición de la cabeza. Este ajuste no debe tocarse si no se trata de un caso de auténtica necesidad, pero algunas veces podemos comprar una cinta comercial que nos cuesta bastante dinero, la probamos y no funciona, la devolvemos a la tienda, nos la cambian por otra, perdemos tiempo y dinero, y finalmente nos damos cuenta de que es nuestro datassette el que no está alineado como los demás. Al igual que con cintas comerciales, nos puede suceder con cintas grabadas en otro datassette, la solución no es demasiado complicada, pero necesita realizarse con cuidado.

Si tenemos el modelo antiguo de datassette (el que era más alto y alargado), la cosa se complica, ya que no permite acceder al tornillo de ajuste desde el exterior, y es necesario desmontar casi todo el datassette para acceder al ajuste. Pero con un poco de suerte tendremos el modelo nuevo (el bajito, cuadrado). En el frente junto al borde de la abertura para introducir la cinta, podemos ver un agujero circular que permite introducir un pequeño destornillador para realizar el ajuste sin tener que abrir la unidad.

Si disponemos de un osciloscopio, podemos colocar la punta de prueba tocando la placa de circuito impreso del datassette a través de un agujero que podemos localizar en la parte inferior de la unidad, a través de él tocaremos la pista que conduce la señal de lectura de la cabeza lectora/grabadora, esta señal es muy pequeña y deberemos ajustar la sensibilidad del canal vertical a unos 20 mV./cm. (la señal debe ser de unos 80 mV. de pico a pico cuando tengamos una cinta grabada colocada en la unidad y el pulsador play presionado). El ajuste deberá realizarse moviendo el tornillo hasta conseguir la máxima señal en el osciloscopio.

Aquellos que no dispongan de instrumental, deberán realizar el ajuste por el "método experimental", girando el tornillo media vuelta en un sentido, probando si la cinta que deseamos cargar entra bien o la cosa va peor, si va mejor pero no del todo giramos otro poco, y así hasta conseguirlo. Si al girar fuese peor que al principio, deberemos girar el tornillo en el otro sentido y probar de nuevo. ■

NOTA IMPORTANTE

Es necesario anotar los movimientos del tornillo que vamos realizando para luego poder dejarlo en el mismo punto que estaba, ya que si lo dejamos muy cambiado no entrarán las cintas que fueron grabadas por nuestro datassette aunque lo hagan correctamente las otras.

Algunas veces es posible dejar el tornillo en un lugar en el que cargan todas las cintas, si fuera posible, ese sería el ajuste correcto, ya que podríamos recuperar los programas que antes teníamos y los nuevos (comerciales o prestados).

Finalmente conviene volver a sellar ese tornillo con una gota de laca de uñas, de este modo no se moverá el tornillo de ajuste.

Espero que estos consejos ayuden a resolver los problemas con los intercambios de cintas y la adquisición de programas comerciales.

¡INCREIBLES TARJETAS PARA COMMODORES 64 Y VIC-20 DE 80 COLUMNAS!

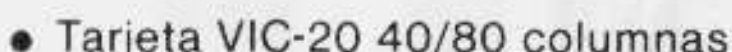
LA TARJETA DE 64K ES LA PERFECTA COMBINACION PARA LA TARJETA DE 40/80 COLUMNAS:

- Puede escoger entre 22/40/80 columnas y obtiene 31743 bytes libres!!!

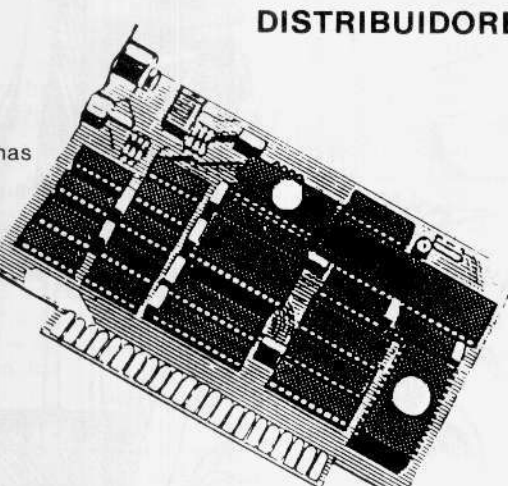
- El sofware del EPROM de la tarjeta le permite utilizar todos los comandos de ficheros (OPEN, CLOSE, PRINT, INPUT, GET, SAVE and LOAD) con los 31232 Bytes de Memoria «oculta», lo que produce la sensación de qué hay 8 cassettes super rápidos conectados a su Vic-20 Imagínese... menos de 1 segundo para guardar 28 K!!!



DISTRIBUIDORES PARA TODA ESPAÑA



- Tarjeta VIC-20 40/80 columnas
- Tarjeta COMMODORE 64 80 columnas
- Tarjeta VIC-20 Ampliación de memoria 64 K - RAM
- Slots de expansión para VIC-20 y 64 de 2 y 5 para conexión de tarjetas



PEDIDOS

FERRE-MORET S.A.

DISTRIBUIDORES
Y PARTICULARES

C/Tusset, 8 - ENTRESUELO 2ª
Teléf.: 218 02 93
08006 BARCELONA

¡ATENCION!

SOFTWARE 80 COLUMNAS

Estos programas requieren las tarjetas distribuidas por FERRE-MORET, S. A. Para el VIC-20, aunque los programas funcionan con un mínimo de 16K, están diseñados para su utilización con la Tarjeta de 64K RAM. Capacidad de almacenaje ilimitada. Todos los programas se venden con un manual explicativo en castellano.

AGENDA DOMESTICA Y DE NEGOCIOS (REF. ADD4080C)

Incluye una función de búsqueda que identifica la totalidad o parte de los nombres, direcciones o números de teléfono, es decir, el sistema permite identificar un nombre, dirección o teléfono, aunque únicamente se recuerde parte de la información. Si el usuario no recuerda una dirección concreta, llamando el nombre de la población, obtendrá todas las direcciones de esa población.

CONTROL STOCK 40 COL. (REF. st40C) CONTROL STOCK COL. (REF. st80C)

Posee una función de búsqueda que permite visualizar productos concretos, especificando el producto, su descripción, lugar de almacenaje, proveedor, dirección del proveedor, código de identificación del producto, precio de coste, precio de venta, cantidad (en paquetes, cajas, etc.), valoración del stock, niveles máximos y minimos de stock. Si además se dispone de impresora, se pueden obtener listados de productos específicos y listados de todo el almacén.

PEDIDOS, ALBARANES, FACTURAS (REF. IDO4080CP)

Se precisa la tarjeta de 64K RAM impresora. El programa permite imprimir encabezamiento con el nombre de la Empresa y su dirección y se puede elegir que el documento sea una Factura, Albarán o Pedido. Otros detalles del encabezamiento son: número de factura, fecha, número de pedido del cliente, referencia del cliente, etc. El programa permite la introducción de varios ITE's, especifica el producto, descripción, cantidad, precio antes de impuestos, precios con impuestos incluidos, gastos de envío etc. Numeración automática de los documentos. Permite hacer copias.

DIETARIO DOMESTICO Y DE NEGOCIOS (REF. HBD80CDP)

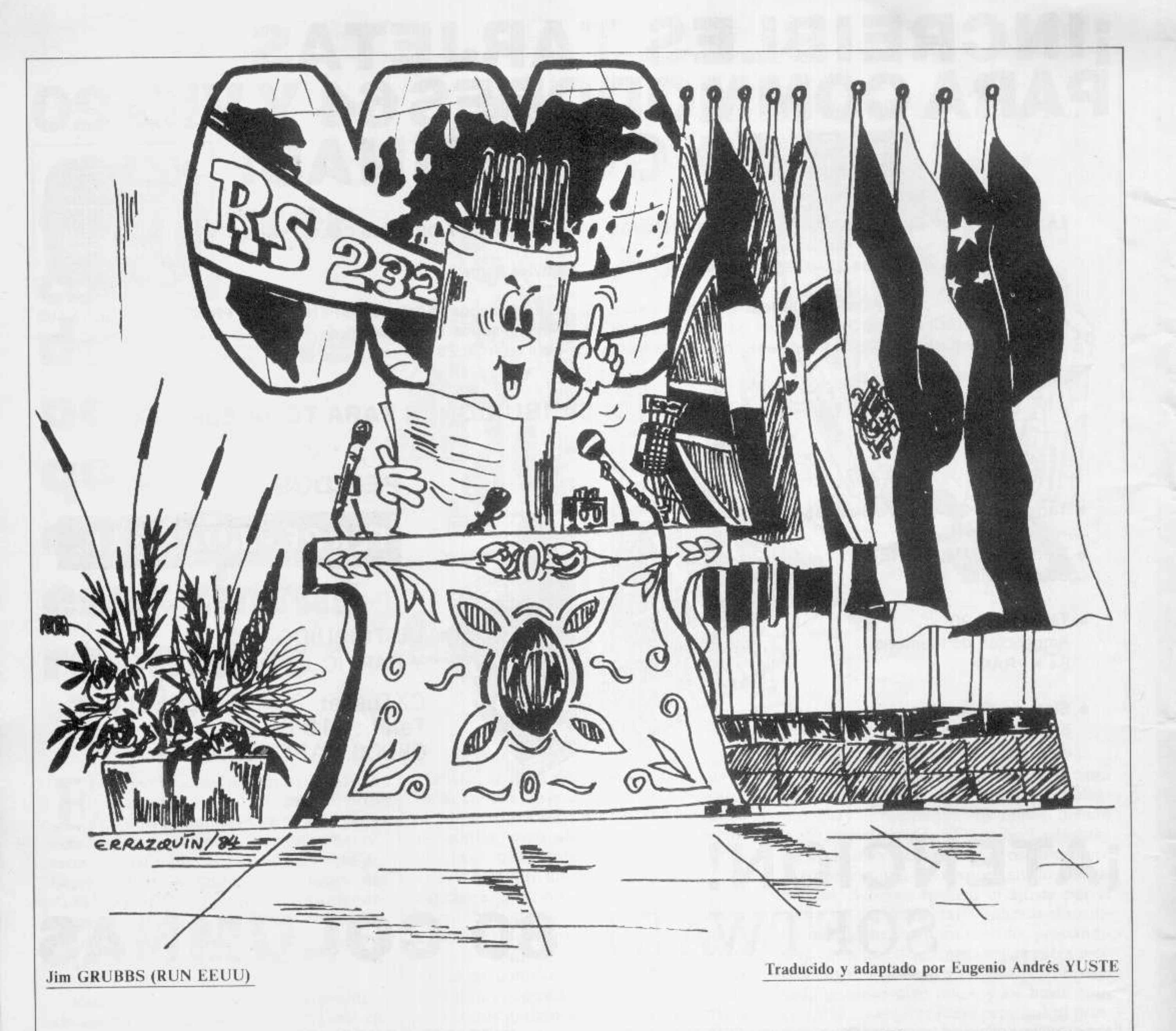
Funciona con Cassette y Unidad de Disco. El programa pide la fecha e identifica cualquier entrevista concertada para ese día, detallando con quién es la entrevista, motivo de la misma, hora y lugar de la reunión, duración de la reunión, etc.
Permite la búsqueda diaria o mensual para identificar reuniones por la fecha, o lugar de reunión, o persona, tanto en el pasado como en el futuro, y además visualiza un calendario.

CONTROL DE ALMACEN A 80 COL. (REF. ST80CDP)

Los datos se pueden almacenar en cassette o Unidad de Disco. Se incluye la descripción de los productos, su referencia o número, cantidad, precio de compra, precio de venta, niveles máximo y mínimo de existencias, pedidos mínimos exigidos por los proveedores, nombre y dirección del proveedor, etc. Se dispone de una función de búsqueda, que permite visualizar todo el almacén, productos cuya cantidad exceda la máxima autorizada, productos con existencia por debajo del nivel mínimo requerido, salidas de almacén, valor del stock, precios y márgenes.

FICHERO DE PERSONAL A 80 COL. (REF. PRF80CDP)

Se necesita impresora. El programa incluye el nombre, dirección, teléfono, edad, fecha de nacimiento, número, sexo, estado, número de Seguridad Social, Cualificaciones, Historial de empleo, Estudios, sueldo, funciones que realiza actualmente, programas de training a los que ha asistido, etc. Se pueden visualizar todos los datos o seleccionar los que se desean, mediante la utilización de una función de búsqueda. Excelente para selección de personal, ya que permite identificación por nombre, pero también se puede buscar por una cualificación o característica concreta, por la edad, experiencia, etc.



Jim GRUBBS (RUN EEUU)

Traducido y adaptado por Eugenio Andrés YUSTE

Pasaporte al mundo (I)

Amplía tus horizontes sobre computadores con este artículo que os abrirá nuevas puertas a tí y a tu Commodore. La clave está en la puerta del usuario de tu Commodore.

Este artículo describe cómo usar tu computador Commodore en una aplicación "real". Esto incluye la conexión de tu computador a aparatos externos. En cualquiera de estas aplicaciones existe el riesgo, por pequeño que sea, de que el aparato externo dañe al computador. Vamos a describir las técnicas de interfaz necesarias, pero si no estás acostumbrado a la electrónica, no intentes conectarlo hasta no conseguir que te ayude alguien que sepa de esto.

Viajando por el mundo descubres cosas muy interesantes. Como, por ejemplo, qué hacer si no sabes ni remotamente de qué forma puedes llegar a tu destino. Se trata de aprovechar los lugares de conexión. Desde uno de esos lugares se te abren las puertas para acceder a mil otros. Supón que, estando en un pequeño pueblo de la provincia de Barcelona, quieres visitar a un amigo que vive en otro pequeño pueblo de la provincia de Madrid. A lo mejor en algún lugar puedes tomar un autobús a un pueblo del de al lado, luego ir en el carro de un campesino, convencer a alguien para que te lleve un poco y andar el resto del camino hasta otro pueblo en que... y no llegarás nunca, claro.

En cambio, hay lugares que son como centros de comunicaciones. Una vez has llegado allí, sabes que te será muy fácil llegar a tu destino. Si quieres ver pronto a tu amigo, sabes que seguramente saldrá de tu pueblo algún autobús con destino a Barcelona. Y, una vez en ese centro de comunicaciones, sabes que puedes llegar a Madrid. Y aunque nunca hayas estado en Madrid, puedes estar seguro de que encontrarás un autobús que te lleve al pueblo de tu amigo. Estos centros de comunicaciones no existen sólo en los transportes, también la compañía telefónica y correos trabajan de esa forma. Incluso tu Commodore 64 o VIC-20 tienen incorporado un centro de comunicaciones, la puerta del usuario.

Da la vuelta a tu computador y obsérvalo por atrás. En el lado derecho verás un conector de doce patas y doble cara, con lo que tienes 24 conexiones en él. A través de este conector puedes contactar con el resto del mundo de los computadores, por ejemplo usando un modem para que tu Commodore hable por teléfono. Además, puedes comunicarte con cualquier aparato que tenga interfaz "RS232" (una cosa muy común aunque quizás te suene nuevo), puedes controlar las luces y tu equipo de música, encender y apagar tu aire acondicionado (si tienes) o la calefacción, tomar datos meteorológicos y hacer mil y una cosas que se te ocurran. Para que puedas controlar todas estas posibilidades tendrás que enterarte un poco de cómo los Commodore leen y escriben en la puerta del usuario. Hay algunos registros importantes que hay que conocer y un poco de aritmética de computadores ayudará bastante. Vamos a echarle un vistazo a esos registros para que puedas conseguir que hagan lo que tú quieres.

Aritmética de registros

El VIC-20 y el Commodore-64 disponen de diversos registros de datos bidireccionales. Esto quiere decir que cada bit del registro se puede usar como entrada o como salida de datos. Incluso el joystick funciona a través de este tipo de registros.

Los registros del VIC-20 se encuentran en las posiciones de memoria 37137 para la puerta A y 37136 para la puerta B. En el Commodore 64 los registros están en la posición 56576 para la puerta A y 56577 para la puerta B. Pero, antes que nada, hay que decirle al computador cómo vamos a usar cada uno de los bits de estos registros (y, con ello cada una de las líneas de datos de la puerta del usuario), si como entrada o como salida).

Para esto escribiremos un número en el registro de dirección de datos (data direction register, DDR en la literatura inglesa) que se encuentra en la posición 37139 (puerta A) y 37138 (puerta B) en el VIC-20 y 56578 (puerta A) y 56579 (puerta B) en el C-64. En este artículo vamos a usar solamente la puerta B de ambos Commodores.

(Recuerda que existe una diferencia significativa entre el C-64 y el VIC-20: el VIC-20 usa como circuito de entrada/salida una VIA 6522 y el C-64 dos CIAS 6526. Esto es importante en cuanto a la puerta de joystick, que es la puerta A en un VIC y la segunda CIA en un C-64. Por ahora, basta con que recuerdes que existe esa diferencia).

No hay una orden especial para ajustar el DDR, si no que (¡cómo no!) hay que "pokear" en el registro el valor adecuado, una vez calculado con aritmética binaria.

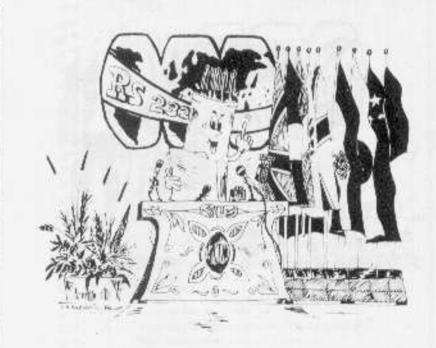
Calma, no te asustes, ni pases al artículo siguiente, ni te vayas a dar de comer al perro. Esto no es nada difícil. Hasta tu profe de matemáticas sabría hacerlo bien. Los DDR, como todas las tripas de tu computador, hablan en binario. Cuando ves la representación binaria de un registro, ves a la derecha el bit menos significativo (least significant bit, LSB) y a la izquierda el bit más significativo (most significant bit, MSB). El registro comprende un solo byte (=8 bits) de memoria, así que numera los ocho bits de 0 a 7, 0 a la derecha, 7 a la izquierda. Para que veas lo fácil que es todo el sistema binario, te voy a explicar cómo se parece al sistema decimal, el sistema de números normal.

Toma un número cualquiera. Por ejemplo 2546.

2546=2×1.000+5×100+4×10+6×1

(¿estás de acuerdo?)

Fíjate en que vamos multiplicando, de derecha a izquierda, cada cifra por diez



veces lo que multiplicábamos la anterior. (1, 10, 100, 1.000...). Esto es así porque éste es el sistema de numeración del 10. Si en decimal las columnas de cifras se multiplican, de derecha a izquierda por 1, 10, 10×10, 10×10×10, 10×10×10×10..., en binario se multiplican por 1, 2, 2×2, 2×2×2, 2×2×2×2, 2×2×2×2×2... Por ejemplo:

1011B=1×1+1×2+0×2×2+1×2×2×2=11D

(hemos empezado esta vez por la derecha)

(B=Binario, D=Decimal)

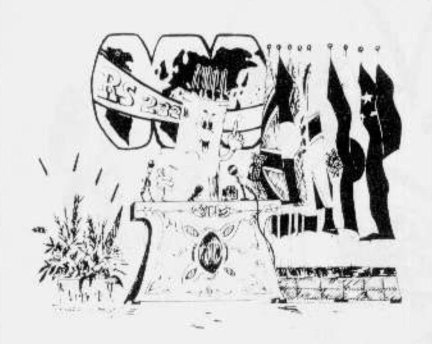
El número en sí es el mismo, pero su representación es diferente. Es lo mismo 1011B pesetas que 11D pesetas, pero una es la forma de pensar del computador y otra es la tuya. Para conseguir que el computador "haga algo" necesitamos muchas veces meter un número en un registro y ese número es fácil saber cuál es en binario. Pero para "pokearlo" hay que dárselo al Commodore en decimal, ya sabes. Entonces sacas tu calculadora y multiplicas y sumas como hemos hecho antes.

Para estos cálculos te será útil la tabla de potencias del 2:

1	=1	=2 ^ 0
2	=2	=2 ^ 1
2×2	=4	=2 ^ 2
2×2×2	=8	=2 ^ 3
2×2×2×2	=16	=2 ^ 4
2×2×2×2×2	=32	=2 ^ 5
2×2×2×2×2×2	=64	=2 ^ 6
2×2×2×2×2×2×2	=128	=2 ^ 7

(así tendrás que multiplicar menos)

El VIC-20 y el Commodore 64 disponen de diversos registros de datos bidireccionales. Esto quiere decir que cada bit del resgistro se puede usar como entrada o como salida de datos. Incluso el joystick funciona a través de este tipo de registros.



Una vez vistas estas reglas, ya podemos calcular los valores a introducir en el DDR para los diversos bits según queramos emitir o recibir datos a través de ellos.

El poli controla el bit

Una vez vistas estas reglas, ya podemos calcular los valores a introducir en el DDR para los diversos bits según queramos emitir o recibir datos a través de ellos.

Vamos a escribir un byte, esto es, ocho bits. Cada bit controla una línea de entrada/salida (una pata del conector). Si el bit vale 0, esa es una entrada, y si vale 1 es una salida. Inicialmente, al enchufar el aparato, todas las líneas son entradas. Si es esto lo que quieres, no tendrás que escribir nada en el DDR. Si quieres tener también salidas, tendrás que escribir en el DDR un número que, puesto en binario, tenga a 1 los bits de las líneas que quieras que sean salidas y a 0 los que quieras que sean entradas. Recuerda: Una entrada → bit correspondiente a 1 en el DDR, una salida → bit correspondiente a 0 en el DDR.

Por ejemplo: De las ocho líneas (las numeramos 0... 7) queremos que sean todas entradas menos la 2 y la 4. Si ponemos una S para salida y una E para entrada, el byte del DDR será:

Nr. de bit	7	6	5	4	3	2	1	0
entrada/salida	E	E	E	S	E	S	E	E
0/1	0	0	0	1	0	1	0	0

Es decir, que tendremos que escribir en el DDR el número binario 00010100B (B=binario). A nuestro Commodore se lo tenemos que contar en decimal, pero ya hemos aprendido a hacerlo. Cogemos los bits uno por uno, comenzando por atrás y vamos multiplicando por los números de la tabla anterior de potencias del dos.

00010100B=
0×1+1+0×2+1×4+0×8+1×16+0×32
+0×64+0×128
=4+16=20

O sea que tenemos que pokear (4+16=20) en el DDR. Como ves, al escribirlo de la forma POKE 37138,(4+16) y no como POKE 37138,20 es más fácil ver que hemos puesto a 1 los bits 2 y 4 (puedes comprobarlo con la tabla del párrafo anterior) y a 0 los demás.

El DDR, o "política de tráfico", hace que la comunicación circule en la dirección correcta. Ya sólo queda escribir en los registros de datos los bits de salida (con un "poke") y leer en los mismos registros de datos el valor de los bits de entrada (con un "peek").

Ya hemos visto como "pokear" para poner los ocho bits de un byte al valor que queramos. Ahora nos queda separar los datos de entrada entre sí. El número que obtienes al hacer un peek contiene, en forma codificada, el valor de los ocho bits de datos. Si sólo deseas saber el valor de uno de estos bits, no tienes más que poner una "máscara" sobre el registro, de forma que sólo veas el bit que deseas comprobar. Esta máscara es como una plantilla que no deja ver más que uno o varios bits seleccionados de los ocho que hay en un byte.

En el lenguaje BASIC del Commodore, la máscara se "pone" con la función AND. La máscara en sí es un número: Es la potencia de dos del número del bit a comprobar. Si, por ejemplo, quieres comprobar el bit 3, de la tabla de potencias del dos ves que el número a usar como máscara es 2^3 = 8. La orden de programación podría ser:

```
IF PEEK (37136) AND 8 =
   1 THEN GOTO 500
```

Esta orden provocaria un salto a la línea 500 si la línea de entrada 3 estuviera a nivel 1.

Libre comunicación

Armado con estos conocimientos, ya puedes comenzar a comunicarte desde tu Commodore 64 o VIC 20 con el mundo exterior. Supón que deseas que el computador ponga en marcha una alarma cuando se abre la puerta principal y que a continuación imprima en pantalla la hora a la que sucedió. Puedes conseguir esto comprobando si está abierta la puerta, para lo que puedes usar un simple interruptor.

Con el mismo método puedes utilizar tu Commodore como temporizador de programas para la televisión o para el video. Escribe un programa que tenga en cuenta todos los acontecimientos que quieras. Después usa unos relés como salida para encender o apagar la televisión o el video automáticamente cuando se cumplan las condiciones que hayas establecido. Los relés los puedes adaptar tú mismo con una etapa amplificadora de un solo transistor, si sabes electrónica; y si no, a lo mejor puedes usar una de las cajas de relés que se venden para el VIC-20 y el C-64 (consulta los anuncios publicados en COMMODORE WORLD).

Si te interesa medir la velocidad del viento, la temperatura, o el nivel de líquido en un tanque, necesitas un convertidor analógico-digital. Si tu aplicación no exije demasiada precisión, puedes usar los convertidores incluidos en tu Commodore, que se han disfrazado habilmente como registros para paddles.

Además te aconsejamos que leas la guia de referencia del programador de tu Commodore 64 ó VIC 20 si deseas más información sobre los circuitos integrados VIA y CIA.

Los relés los puedes adaptar tú mismo con una etapa amplificadora de un solo transistor, si sabes electrónica; y si no, a lo mejor puedes usar una de las cajas de relés que se venden para el VIC-20 y el C-64.

Contabilidad 64

Versión B
Precio: 29.500
Autor: José Tinto Espelt
Propietarios y distribuidores:
Casa de Software S. A.
Aragón, 272, 8, 6
08007 Barcelona
Tel. (93) 215 69 52

Este paquete de contabilidad se compone de un cartucho y un disco. La configuración mínima para su funcionamiento es un C-64 ó SX-64, una unidad de disco (ya incorporada en el SX), y una impresora.

El primer punto a favor de este paquete es que puede utilizar muchos tipos de impresoras, desde las Commodore VC1515 y MPS801, a las GP-100, las EPSON y otras, con lo que quien prefiera utilizar otro tipo de impresora, que se adapte más a sus necesidades, puede hacerlo siempre y cuando la impresora pueda utilizar el protocolo Centronics.

Entre las especificaciones del programa destacamos que puede controlar 600 cuentas con 2.300 apuntes por diskette, y siempre se puede enlazar un disco con otro, pues los resultados anteriores son transferidos al nuevo disco que se utiliza.

No soy un experto contable, pero con ayuda de amigos que lo son y conocen bien el tema, hemos utilizado el paquete y nos pareció muy bien y muy completo. Cualquier operación contable está tenida en cuenta. El manual bien presentado, conciso pero claro y perfectamente explicado. Mantenimiento de ficheros, listado de ficheros y utilitarios.

Por otro lado, presenta la buena idea de dedicar un apartado completo a Nociones de Contabilidad para los "novicios", para quienes la contabilidad puede representar un cierto misterio. Igualmente la última sección del manual dedicada al Plan General Contable Español es una valiosa ayuda para expertos y novicios.

El cartucho contiene el sistema de arranque del programa, son 8 K de EPROM que contienen unas rutinas en código máquina y el programa cargador en basic.

casa
de
software sa

CONTABILIDAD
64

600 cuentas
2.000 apuntes
(por diskette)

impresora paralelo cetronics

CONTIENE:
Manual de usuario
Nociones fundamentales de contabilidad
Plan General Contable Español

PARA EL ORDENADOR
COMMODORE 64

Cuando encendemos el ordenador pide que pongamos el disco, lo carga y arranca. Los programas están escritos en basic, pero con rutinas en código máquina, de este modo aprovecha las ventajas de los dos (sencillez y velocidad a la vez). En realidad son varios programas, uno contiene el menú principal, y los otros desempeñan las diferentes funciones (asientos, diarios, consultas de cuentas, extractos, sumas, saldos, situación y cierre, cuentas de explotación, balances, mantenimiento de ficheros, listado de ficheros y un grupo de utilitarios).

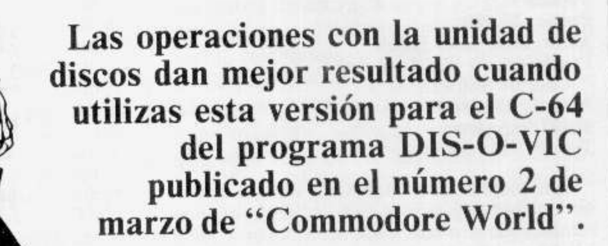
Me pareció interesante la incorporación de los utilitarios que permiten copiar el disco, recuperar ficheros, etc. Expecialmente pensando en los datos que se pierden a veces en las

(Pasa a pág. 66)

Cal OVERHULSER (RUN EEUU)



Traducido por Valerie SHANKS y Adaptado por Diego ROMERO



Las operaciones con la unidad de discos dan mejor resultado cuando utilizas esta versión para el C-64 del programa DIS-O-VIC publicado en el número 2 de marzo de "Commodore World".

Disk-O-64

Añade 13 Comandos Relacionados con Disco a tu C-64

En el número de marzo de "Commodore World" publicamos un programa de utilidades para disco para el VIC-20 llamado DISK-O-VIC. Resulta ser un programa muy útil que facilita mucho el manejo de la unidad de discos con el VIC-20. Yo quería conseguir lo mismo para el C-64, y por lo tanto decidí convertir el programa DISK-O-VIC en DISK-O-64.

El principal problema era convertir las direcciones de las llamadas al sistema del DISK-O-VIC en las direcciones adecuadas para el DISK-O-64. Las llamadas Kernal resultaban sencillas, dado que son las mismas para los dos equipos y se han publicado en varios manuales de referencia. El verdadero problema eran otras llamadas al sistema como Warmstart, Reset y Printstring.

Busqué en las ROM del C-64 para encontrar las rutinas que necesitaba. La Tabla 1 presenta los nombres de variables del listado original del DISK-O-VIC que necesitan cambiarse, junto con las nuevas direcciones del sistema para el C-64. Una vez conseguidas las direcciones correctas, la conversión en sí resultó bastante sencilla.

Primero, encontré las llamadas que se tenían que modificar en los mismos sitios que se situaban en el listado original y localicé sus equivalentes en el volcado hexadecimal. Después, tuve que establecer los cambios necesarios para la pantalla de arranque. A continuación cargué el programa DISK-O-VIC, realicé los cambios necesarios con un monitor y salvé una copia de DISK-O-64 con la misma longitud y las mismas capacidades que DISK-O-VIC.

Entrar el programa

Se necesita un monitor de lenguaje máquina para entrar el programa DISK-O-64 del listado del volcado hexadecimal. Después de cargar y entrar el monitor, se empieza a entrar el programa en la dirección $0801 y se continúa hasta $0D2F. A continuación utiliza el método adecuado para salvar DISK-O-64 en tu monitor. Si utilizas el monitor del C-64 de Commodore, tienes que teclear:

S"DISKO64",08,0801,0D2F

Asegúrate de que hayas utilizado $0801 como la dirección inicial para que lo puedas cargar después como cualquier programa en Basic. Ahora dispones de una copia de DISK-O-64 que puede ser cargado y salvado como cualquier programa en Basic. A continuación, sales del monitor, tecleas SYS64738 en el modo Directo, o bien apagas y enciendes el ordenador.

Ahora puedes cargar y ejecutar DISK-O-64 igual que cualquier otro programa en Basic. Simplemente teclea LOAD "DISK-O-64",8 para cargar el programa en el ordenador y a continuación teclea RUN. Si todo funciona como es debido, aparece la pantalla de arranque, y el programa DISK-O-64 se encuentra en su sitio, protegido en la parte superior de la memoria.

La Tabla 3 presenta los nuevos comandos. Si lees el artículo original sobre DISK-O-VIC, verás que todos los comandos son iguales para el DISK-O-64. Utiliza un disco para experimentar y prueba cada comando para poder familiarizarte con ellos (además de asegurar que todo funciona bien, y no existen errores de teclear).

Modificaciones de DLoad/DSave

Había una cosa que me molestaba en el programa original DISK-O-VIC. No podía utilizar DLoad/DSave en un programa híbrido (es decir, un programa escrito en Basic y lenguaje máquina).

Parecía que se cargaba y se salvaba correctamente, pero me di cuenta de que el programa salvado tenía menos bloques que el programa original.

El verdadero problema era que no me daba cuenta de lo que estaba pasando hasta que un día el programa híbrido DISK-O-VIC no quería ejecutarse, y lo tuve que volver a teclear todo. Con todo hay que decir que esta restricción fue mencionada en el artículo, y el fallo fue mío al no darme cuenta.

Descubrí que el problema venía de la rutina DLoad en el programa DISK-O-VIC. Esto ha sido modificado en el programa DISK-O-64 para que DLoad/DSave puedan ser utilizados con programas híbridos con tal de que se carguen como un programa normal en Basic, es decir, LOAD"NAME",8.

La única excepción es el comando Append, que puede ser utilizado para añadir solamente programas escritos en Basic, y no programas híbridos. En realidad, la necesidad de añadir las partes escritas en Basic de programas híbridos no es muy frecuente (yo no lo he hecho nunca).

La Tabla 2 presenta los cambios para editar el programa original DIS-O-VIC para los que quieran modificar sus copias del programa.

Tabla 1

Modificaciones de las rutinas para el DISK-O-64 sobre el listado del DISK-O-VIC

RUTINA	VIC	C-64
WARMST	C474	E386
WAIT	C48C	A48C
INFIN	C49F	A49F
CHAIN	C533	A533
CLR	C659	A659
INTEGR	C96B	A96B
PSTRNG	CB1E	AB1E
ERROR	CF08	AF08
PRLINE	DDCD	BDCD
CHROUT	E742	E716
RESET	FD22	FCE2

Tabla 2

Modificaciones en el programa DISK-O-VIC para los comandos DLoad/DSave con programas híbridos.

DIRECCION HEXADECIMAL DEL VIC	NUEVO VALOR HEXADECIMAL
126E	86
126F	2D
1270	84
1271	2E
1272-127B	EA
127C	20
127D	33
127E	C5

Tabla 3

Explicación de los Comandos del programa DISK-O-64.

APPEND (Agregar).—Este comando permite que un programa en Basic que se encuentra en disco se agregue al final de un programa en memoria con números de línea más bajos. Su sintaxis correcta es: "APPEND "NOMBRE".

CATALOG (Catalogar).—Tecleas CATALOG para poder leer el directorio contenido en el disco sin destrozar el programa en la memoria.

COLLECT (Recoger).—Esto realiza una operación de validación, es decir, que ordena el disco y hace que todos los bloques no utilizados sean disponibles.

DLoad.—Esto funciona igual que el comando normal de Load, pero no hace falta teclear ,8. Además, antes de cargar, inicializa el disco, y después lo verifica. Se pueden cargar tanto los programas en Basic como los programas híbridos siempre y cuando se carguen los programas híbridos de la misma forma que los programas en Basic.

DSave.—Igual que DLoad, sino que su función es la de salvar los programas en disco.

HEADER.—Este comando formatea el disco. Dado que todos los datos serán destruidos, te pregunta "ESTAS SEGURO? SI O NO". El sintaxis correcto es: HEADER"NOMBRE NUEVO",Ixx. Tienes que emplear ,I. Las letras xx representan cualquier tipo de identificación que le quieras poner (uno para cada disco).

INIT.—Esto es igual que OPEN15,8, 15,"I":CLOSE15 en Basic.

KILL.—Esto es igual que apagar y volver a encender el ordenador o teclear SYS64738.

OFF.—Este comando apaga el DISK-O-64, pero este y cualquier otro programa quedan en la memoria. El programa DISK-O-64 hace que el Basic vaya más despacio, así que es mejor quitarlo si te interesa la máxima velocidad. Para volver a encenderlo, teclea: SYS256*PEEK(56)+PEEK(55).

RENAME.—Te permite volver a nombrar un programa que ya existe en disco. Su sintaxis correcta es: RENAME"NOMBRE ANTIGUO"TO"NOMBRE NUEVO".

SCRATCH.—Con este comando se puede borrar un programa del disco; su equivalente en Basic es OPEN15,8,15,"S0-:NOMBRE":CLOSE15. También te pregunta "ESTAS SEGURO? SI o NO". Su sintaxis correcta es: SCRATCH"NOMBRE".

SEND (Enviar).—Este comando te permite enviar al disco cualquier comando que se pueda enviar en Basic; su equivalente en Basic es OPEN15,8,15,"xxxx":CLOSE15, donde xxxx representa el mensaje que envías. Su sintaxis correcta es: SEND "xxxx".

STATUS (Estado).—Con este comando se puede visualizar el estado del disco sin tener que ejecutar un programa. Cuando surge un error en disco, teclea STATUS. ■

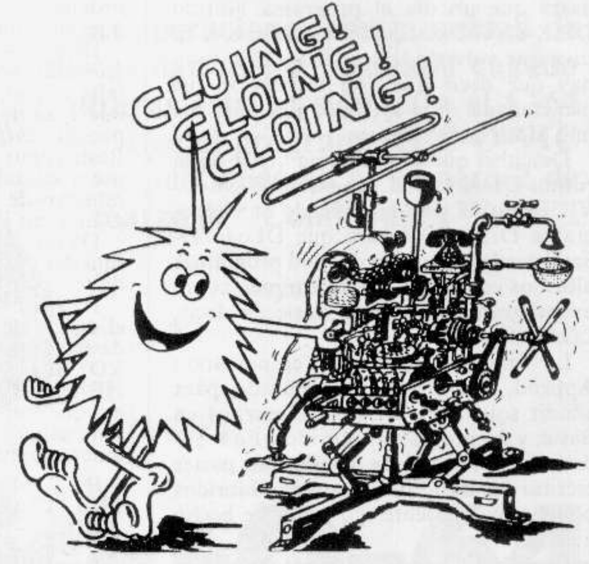
Listado del volcado hexadecimal

```
.:0800 00 40 08 01 00 99 22 93
.:0808 11 12 22 A3 31 35 29 22
.:0810 44 49 53 4B 2D 4F 2D 36
.:0818 34 11 22 A6 32 33 29 22
.:0820 43 36 34 20 44 49 53 4B
.:0828 20 43 4F 4D 4D 41 4E 44
.:0830 20 50 41 43 4B 41 47 45
.:0838 11 22 A6 32 30 29 3B 00
.:0840 7A 08 02 00 99 22 42 59
.:0848 20 43 41 4C 20 4F 56 45
.:0850 52 48 55 4C 53 45 52 11
.:0858 22 A6 31 38 29 22 46 52
.:0860 4F 4D 20 44 49 53 4B 4F
```

(Pasa a pág. 53)



MANUAL DEL MONITOR DE CODIGO MAQUINA PARA EL C-64 HES-MON (II)



G=Ejecución

Empieza la ejecución de un programa en código máquina a partir de la dirección almacenada en el registro PC (contador de programa).

Ejemplos:

.G

El HESMON va a la dirección que indica el Contador de Programa PC (ver R) y empieza la ejecución del código máquina allí existente.

.G 3000

El HESMON pone la dirección $3000 en el PC y empieza la ejecución del código máquina a partir de esta dirección.

H=Busca en memoria

Búsqueda de bytes que tengan cierto valor. El HESMON visualiza las direcciones que contienen un valor (o valores) determinado en una zona de memoria concreta.

Ejemplos:

.H C000 D000 20 D2 FF

El HESMON chequea los bytes de memoria comprendidos entre $C000 y $D000 buscando la secuencia de bytes $20, $D2 y $FF (JSR $FFD2 en lenguaje ensamblador). Si encuentra tres bytes consecutivos que cumplan esta condición, visualiza la dirección del primero de ellos y continúa buscando, hasta llegar al límite indicado (en este caso $D000).

.H C000 D000 "READ"

El HESMON chequea los bytes de memoria comprendidos entre $C000 y $D000 buscando la secuencia en caracteres ASCII correspondientes a "READ", visualizando las direcciones en donde la encuentre. La búsqueda puede interrumpirse pulsando la tecla STOP.

I=Contenido ASCII

Visualiza el contenido en hexadecimal y en ASCII de una zona de memoria. Por ejemplo:

.I F000 F020

Si no se especifica dirección final (segundo parámetro) el HESMON visualiza hasta el final de la memoria ($FFFF) o hasta que se pulse la tecla STOP.

J=Salto a subrutina

Ejecuta una llamada a una subrutina y vuelve al programa principal con el modo W desactivado —durante la ejecución de la subrutina— sólo cuando está en modo W. Supongamos que estamos en modo W y en la pantalla se muestra el fragmento de código:

```
147F LDA #$00
1481 JSR $A2C7
J
```

Se ejecuta la rutina que se inicia en $A2C7. Cuando encuentra un RTS, volverá al modo W.

L=Load

Carga de zonas de memoria desde disco o cassette. Ejemplos:

.L "RAM TEST" 01

Carga desde el cassette el programa llamado "RAM TEST".

.L "O:RAM TEST" 08

Carga desde el drive 0 del periférico 8 el programa llamado "RAM TEST". Los programas se cargan en la misma dirección de memoria en la que estaban cuando fueron grabados.

M=Contenido hexadecimal

Visualiza el contenido de una zona de memoria en formato hexadecimal. Los bytes visualizados pueden ser modificados directamente situando el cursor sobre ellos, efectuando la modificación, y pulsando RETURN. Por ejemplo:

.M 1000 1005

```
.: 1000 43 0D 0A 43 42
.: 1005 0D 0A 7F 7F 7F
```

N=Transferencia de contenido

Traslada código máquina de una zona de memoria a otra. Se le puede indicar que las instrucciones de tres bytes, que apuntan a direcciones de memoria absolutas (JSR $FCCO, CMP $7540, X...) cambien su valor adaptándolo a la nueva posición que ocupen, o bien conserven inamovible el que ya tienen.

Ejemplos:

.N 7000 77FF 1000 0400 8000

Traslada el código máquina comprendido entre $7000 y $77FF a partir de la dirección $1000 (hasta $17FF). Los dos últimos parámetros indican que las instrucciones de tres bytes que apunten a una dirección comprendida entre $0400 y $8000 deben cambiar su valor y adaptarse a la nueva posición del código máquina, por lo que en este caso se le restará $6000 a cada una. Las instrucciones de tres bytes que apunten a otra dirección que no sea la comprendida entre $0400 y $8000 se trasladarán manteniendo intacta su dirección.

Este comando se detiene cuando encuentra un código de ensamblador erróneo (lo visualiza para indicar dónde se ha parado).

.N 7000 77FF 0400 8000 W

El parámetro "W" al final indica que se está trasladando una tabla de bytes .WORD, por lo que todas las direcciones (tomadas de dos en dos bytes) cuyo valor esté comprendido entre $0400 y $8000 se adaptan a la nueva posición de memoria.

O=Desviación de salida

Este comando es el equivalente en MLM del comando BASIC CMD. Este permite al HESMON desviar la salida de pantalla hacia la impresora o hacia la unidad de disco.

Por ejemplo:

.008 02 "O:TEST"

Donde 08 es el número de periférico (04 para la impresora) y 02 es la dirección secundaria (normalmente 02 a 0E para la unidad de disco).

Todos los parámetros son opcionales. Si Vd. escribe sólo O automáticamente



HESMON abrirá el canal de impresora. Si vuelve a escribir O, volverá a las condiciones normales, cerrando el canal de comunicación con la impresora.

P=Copia de pantalla

Este comando es una versión limitada del comando O. Copia el contenido de la pantalla en la impresora. Los parámetros de este comando son los mismos que para O, excepto que se desactiva automáticamente porque copia exactamente el contenido de la pantalla.

NOTA: P no copiará gráficos de Alta Resolución o Sprites.

Q=Ejecución con puntos de ruptura

Empieza la ejecución de un programa en código máquina a partir de la dirección que indica el PC (contador de programa). La ejecución es más lenta que la del comando, G, pero el Q puede detenerse mediante puntos de ruptura habilitados por el comando B.

Ejemplos:

.Q

Inicia la ejecución del código máquina a partir de la dirección que indica el Contador de Programa (PC).

.Q 1000

Asigna al PC el valor $1000 y empieza la ejecución desde esta dirección.

R=Contenido de registros

Visualiza el contenido de los registros según la siguiente clave:

PC: Contador de programa.
SR: Registro de estado.
AC: Acumulador.
XR: Registro índice X.
YR: Registro índice Y.
SP: Puntero del stack.

Los valores de los mismos pueden ser cambiados situando el cursor sobre ellos y pulsando RETURN.

Ejemplo:

.R

```
      PC  SR  AC  XR  YR  SP
.;  0000  01  02  03  04  05
```

Puede usarse este comando para asignar los valores del contador de programa, acumulador e índices antes de ejecutar programas mediante G, Q o W.

RB=Quitar puntos de ruptura

Eliminar los puntos de ruptura establecidos con el comando B. Solamente hay que especificar el punto de ruptura y será eliminado.

Por ejemplo:

.RB 1050

El punto de ruptura existente en la posición $1050 ha sido eliminado.

S=Save

Grabación en disco o cassette de zonas de memoria. Ejemplos:

.S "PROGRAMA" 01 0800 0C81

Graba en el cassette, bajo el nombre de "PROGRAMA", el contenido de los bytes comprendidos entre las direcciones $0800 y $0C80. Obsérvese que la última dirección a escribir en el periférico es la última **más uno** en el comando.

.S **"O:PROGRAMA" 08 0800 0C81**

Graba en el drive 0 de la unidad de discos cuyo número de periférico es 8, bajo el nombre de "PROGRAMA", el contenido de los bytes comprendidos entre las direcciones $800 y $0C80.

T=Transferencia de memoria

Traslada el contenido de una zona de memoria de un lugar a otro de la RAM. Por ejemplo:

.T 1000 1100 5000

Toma los bytes comprendidos entre $1000 y $1100 y los sitúa a partir del $5000.

Este comando no cambia ninguna dirección aunque se traslade código máquina, a diferencia de N.

U=Test de la RAM de color

U no necesita parámetros. Prueba la RAM de color e imprime OK cuando ha ejecutado el test. Si detecta un byte defectuoso, imprimirá su dirección.

V=Test de RAM

Este comando prueba un segmento de RAM. HESMON irá comprobando cíclicamente la sección de RAM especificada hasta que se pulse la tecla STOP (puede ser necesario pulsarla más de una vez). Cada vez que HESMON completa el test de la sección de RAM en pruebas, imprimirá un "." para que pueda ver que el comando está en acción.

Si detecta un fallo en una posición de memoria, imprimirá la posición de memoria seguida de un número binario que muestra el dato que no ha sido guardado correctamente. Los bits de este número se muestran de izquierda a derecha (del bit más significativo al bit menos significativo). Los bits que presentan problemas se muestran con inversión de color.

Por ejemplo:

.V 0400 0800

Verificará la RAM entre las direcciones $0400 y $0800 hasta que pulse la tecla STOP o ocurra un fallo.

W=Ejecución paso a paso

Ejecuta una instrucción de código máquina a partir de la dirección indicada por el PC, mostrando el contenido desensamblado de la próxima instrucción a ejecutar. Al mismo tiempo también muestra el valor de los registros (R).

Ejemplos:

.W

Ejecuta la instrucción apuntada por el PC, muestra los valores de los registros, visualiza el desensamblado de la siguiente instrucción, y aguarda a que se pulse una tecla para ejecutarla. En el caso de que se pulse una tecla que normalmente tiene repetición (la barra espaciadora, por ejemplo) la ejecución se realizará al ritmo de esta repetición. **.W.1000**

Pone la dirección $1000 en el PC y ejecuta la instrucción, muestra los valores de los registros, visualiza el desensamblado de la siguiente instrucción, y aguarda antes de ejecutarla.

X=Salida al Basic

Sale del HESMON y pasa el control de C-64 al intérprete BASIC. Este comando tiene dos formas de actuación: La primera "XC" tiene el mismo efecto que desconectar el equipo excepto que el HESMON continúa conectado y puede transferírsele el control pulsando la tecla RESTORE. La segunda forma —"X"— se comporta de forma similar a la pulsación simultánea de las teclas RUN/STOP y RESTORE en BASIC, no obstante **su primera salida del HESMON debe hacerla con "XC", de lo contrario el BASIC funcionará mal**. Una vez en BASIC para conseguir el mismo efecto que con RUN/STOP y RESTORE sin HESMON, pulsar RESTORE, pulsar "X" y RETURN.

Conversión decimal a hexadecimal

Este comando imprime el equivalente hexadecimal de un número decimal. Si el número decimal es negativo mostrará el complemento a dos de un número de 16 bits hexadecimal y el número decimal positivo correspondiente.

Por ejemplo: **.# 1234**

HESMON contestará en la misma línea: .#1234= $04D2 1234.

$=Conversión hexadecimal a decimal

Este comando imprime el equivalente decimal de un número hexadecimal.

Por ejemplo: **.$ ABCD**

HESMON mostrará (en la misma línea): .$ ABCD 43981

+ = Suma hexadecimal

Este comando imprime la suma de dos números hexadecimales en hexadecimal y decimal. Se deben usar cuatro dígitos hexadecimales.

Por ejemplo: **.+ 1234 5678**

HESMON muestra:
1234 5678 = $68AC 26796

—=Resta hexadecinal

Este comando imprime la diferencia entre los dos números hexadecimales en hexadecimal y decimal.

Por ejemplo: **.—1234 5678**

HESMON mostrará:
-1234 5678 = $BBBC 48060

Como aclaración final a la lista de comandos del HESMON debe tenerse en cuenta que los comandos que producen listas en pantalla (D, I y M) permiten que mediante las teclas de movimiento de cursor vertical se pueda prolongar el efecto del comando en las direcciones anteriores (con la tecla SHIFT) o posteriores a la última con la que se ha trabajado.



M. STOLLENWERK - Frankenberger Str. 32. 5100 AACHEN R.F. de Alemania.

EL VIC-20 POR DENTRO



Algo de soft...

Una de las rutinas más importantes y más sencillas del ROM del VIC-20 es la rutina del interpretador de BASIC. Aquí se va a comentar de tal manera que no habrá apenas dificultades para implementar nuevas instrucciones al BASIC.

La rutina CHRGET para coger el siguiente carácter del programa se ha puesto en la primera página de la memoria ($0000-$0100) porque así se ejecuta más rápidamente. Al encender el aparato se copia la rutina del ROM al RAM.

```
C7AE     20 2C C8  JSR $C82C       comprueba tecla STOP
C7B1     A5 7A     LDA $7A         pointer del programa
   3     A4 7B     LDY $7B            "      "     "
   5     C0 02     CPY #$02        ¿Modo directo?
   7     EA        NOP
   8     F0 04     BEQ $C7BE
   A     85 3D     STA $3D         no: salvar pointer para CONT
   C     84 3E     STY $3E
   E     A0 00     LDY #$00
C7C0     B1 7A     LDA ($7A),Y     coger carácter actual de prg.
   2     D0 43     BNE $C807       ¿final de linea?
   4     A0 02     LDY #$02
   6     B1 7A     LDA ($7A),Y
   8     18        CLC             poner flag para END
   9     D0 03     BNE $C7CE       ¿final prg?
   B     4C 4B C8  JMP $C84B       si: ejecutar END
   E     C8        INY
   F     B1 7A     LDA ($7A),Y
C7D1     85 39     STA $39         memorizar no. de linea actual
   3     C8        INY
   4     B1 7A     LDA($7A),Y
   6     85 3A     STA $3A
   8     98        TYA
   9     65 7A     ADC $7A         pointer prg. a la siguiente linea
   B     85 7A     STA $7A
   D     90 02     BCC $C7E1
   F     E6 7B     INC $7B
C7E1     6C 08 03  JMP ($0308)     JMP $C7E4
   4     20 73 00  JSR $0073       CHRGET coge siguiente carácter
   7     20 ED C7  JSR $C7ED       ejecutar instrucción
   A     4C AE C7  JMP $C7AE       al principio de rutina

******************************     decodificar instrucción
C7ED     F0 3C     BEQ $C82B       si final linea RTS ($C82B)
   F     E9 80     SBC #$80
C7F1     90 11     BCC $C804       code del interpretador
   3     C9 23     CMP #$23
   5     B0 17     BCS $C80E       code de función o GO TO
   7     0A        ASL
   8     A8        TAY
   9     B9 0D C0  LDA $C00D,Y
   C     48        PHA             coger dirección de instrucción
   D     B9 0C C0  LDA $C00C,Y     de la tabla
C800     48        PHA
   1     4C 73 00  JMP $0073       coger siguiente carácter y eje-
                                   cutar instrucción
   4     4C A5 C9  JMP $C9A5       a instrucción LET

******************************     comprobar siguiente instrucción
C807     C9 3A     CMP #$3A        ':'
C809     F0 D6     BEQ $C7E1
   B     4C 08 CF  JMP $CF08       'SYNTAX ERROR'

******************************     comprueba code GO TO
C80E     C9 4B     CMP #$4B        'GO'-code menos 80
C810     D0 F9     BNE $C80B
   2     20 73 00  JSR $0073       CHRGET coge siguiente carácter
   5     A9 A4     LDA #$A4        'TO'-code
   7     20 FF CE  JSR $CEFF       comprueba code
   A     4C A0 C8  JMP $C8A0       a la instrucción GOTO
```

Algo de hard...

El circuito integrado más interesante del VIC-20 es el Video Interface-Chip.

Para obtener resultados del ordenador hace falta una unidad de salida. El VIC-20 usa normalmente un televisor de color. Todas las señales necesarias para su funcionamiento las genera el VIC, el Video-Interface-Chip 6561. La única excepción es la señal de alta frecuencia a la que se modula la información. Esta frecuencia la genera el modulador externo. Todas las otras señales vienen del VIC.

Además posee este circuito integrado (IC) de 40 patillas dos conversores analógicos/digitales, cuatro generadores de frecuencia independientes y programables y una entrada para un Light-Pen. Pero primero vamos a ver el significado de cada patilla:

Pin	Desc.	Función
1	N.C.	sin función
2	COMPCOLOR	salida, información sobre color para TV.
3	SYNC&LUM	salida, información sobre sincronización y brillo para TV.
4	VR/W	entrada, lectura/escritura
5	VD11	bit de datos 11
hasta		
8	VD8	bit de datos 8
9	BD7	bit de datos 7
hasta		
16	BD0	bit de datos 0
17	POT X	entrada, conversor A/D 1
18	POT Y	entrada, conversor A/D 2
19	COMP SND	salida sonido
20	Vss	alimentación masa
21	A0	bit de dirección 0
hasta		
34	A13	bit de dirección 13
35	P01	salida ciclo
36	P02	salida ciclo, sin usar
37	LITE PEN	entrada lápiz óptico
38	02in	entrada frecuencia de cuarzo
39	01in	entrada frecuencia de cuarzo
40	Vdd	alimentación +5V



Al observar las patillas del VIC resalta el bus VD8 hasta VD11, con lo que dispone de un bus de datos de 12 bits. Esto resulta algo extraño. Pero la respuesta es tan fácil como convincente. Los cuatro bits adicionales sirven para definir el color del carácter. Y así también se explica la posibilidad de 16 colores, porque con 4 bits se pueden hacer 16 combinaciones (...).

Las restantes 8 líneas de datos van al bus de datos BD. Este bus tiene una doble función. Cuando el ciclo del sistema está en H lo usa el procesador para acceder al RAM. En el estado L lo usa el VIC. Ahora el VIC lee el Vídeo —y color— RAM, y salva los datos necesarios para la representación de los caracteres del Character-ROM.

También las líneas de direcciones del VIC tienen una doble función. En el tiempo que el procesador ocupa el bus de direcciones también direcciona a los 16 registros internos del VIC.

Las entradas POT X y POT Y están conectadas a las patillas 5 y 9 del Controller-Port para el scanning de los Paddles. Su resistencia tiene que variar entre 200 y 500.000 ohmios. La entrada del lápiz óptico va directamente a la patilla 6 del mismo port.

La señal de baja frecuencia de los cuatro generadores (patilla 19) se filtra para poder acceder a ella por el contacto 3 del Vídeo/Audio-Port. Se tiene la posibilidad de mezclar otra señal de audio, p. ej., un micrófono, que sólo tiene que conectarlo al contacto Y del Expansion-Port y fijarse de que la señal tenga unos 3 Vss.

COMPARTIENDO EXPERIENCIAS ENTRE AMIGOS



Esta sección está dedicada a la colaboración de todos nuestros lectores y está dividida en dos partes:

1) Programación: Programas y similares.

2) Magia: Trucos, sugerencias, etcétera (pág. 48)

Habrá premios y alicientes para todos los participantes (ver pág. 5).

Enviarnos vuestra dirección para que podáis poneros en contacto unos con otros. ¡Animaros, chicos (...y chicas)!

Sabotaje Nuclear

C-64

Francisco Sarrias Pedemonte
Laforja, 100, 2º 3ª e-B
Barcelona-21

El grupo terrorista MOPILETTE ha saboteado los silos de almacenamiento de plutonio concentrado. Estos silos si estallan provocan una reacción en cadena que destruiría la humanidad.

Su misión como jefe de seguridad de la central consiste en penetrar en las distintas habitaciones del complejo y cerrar las válvulas numeradas según el orden que indiquen las claves de desactivación.

Pero atención: si desactiva las válvulas en un orden que no es el indicado, usted es destruido y las esperanzas de la humanidad se desvanecen.

Una advertencia final: las habitaciones contienen gases letales y usted ha de consumir el oxígeno que hay en su vehículo que va disminuyendo con rapidez y si llega a extinguirse usted perece.

Descripción del Juego:

Al comenzar el juego se encuentra usted en su vehículo esférico en una enorme sala donde están marcadas las válvulas de seguridad. En la parte inferior de la pantalla está indicado el orden en que usted debe pasar por los números para desactivar las cargas situadas en esa habitación. Para mover su vehículo utilice las siguientes teclas:

Y - arriba   N - abajo
G - izquierda   J - derecha

Al comenzar la partida, usted debe seleccionar su nivel de juego que determinará la rapidez con que su reserva de oxígeno se consumirá, así como la puntuación de BONUS que usted reciba al finalizar la pantalla.

Al cerrar una válvula el ordenador comprueba si ésta seguía la cadencia indicada en la clave de la habitación; de ser así ese número se coloca en rojo en la clave y usted procede a pasar por el siguiente número; de lo contrario usted es destruido y pierde una vida de las tres que se le conceden al iniciarse el juego.

Otra forma de perder la vida es cuando se le agota la reserva de oxígeno —que inicialmente es de 999 litros—; cuando el oxígeno es muy escaso su marcador de oxígeno se volverá loco.

Tanto el nivel como las vidas que se tienen se incrementan automaticamente a medida que se va adquiriendo una cierta puntuación, así pues el nivel se aumenta más o menos cada 5.000 puntos —digo más o menos porque igual aumenta 2 ó 3 unidades de golpe, forma parte del riesgo y la aventura... y las vidas aumentan cada 15.000 puntos.

Según el nivel en que se halle el oxígeno disminuirá más o menos lentamente y usted tendrá que dirigir su vehículo con mucha habilidad para aprovecharlo al máximo —piense que su vehículo esférico es un modelo anticuado y de velocidad limitada.

Al conseguir cerrar todas las válvulas de una sala, su puntuación se verá incrementada —BONUS— en función del nivel y del oxígeno que aún le resta.

Sólo me queda advertir que yo personalmente aún no he pasado del nivel seis...

Descripción del Programa:

Si bien el programa no goza inicialmente de una espectacularidad desbordante, resulta bastante adictivo y convida al jugador a probar de superar su anterior puntuación.

El programa carece de efectos de sonido y de otros utillajes de costumbre en la "full arcade action" tales como la utilización de rutinas en "machine code" para dar rapidez a los movimientos. Pero mi objetivo era dar una idea básica que contuviera los menos "pokes" posibles y que fuera rápida y sencilla de copiar. Sería interesante observar los adornos que cada cual añada al juego y comparar los resultados finales.

Algunas implementaciones sencillas son la aparición de minas, habitaciones creadas por random, utilización de "esprites" en los gráficos, sonidos, tablas de records, paredes de la habitación electrificadas, monstruos que te persigan, etc... cada cual con su imaginación y habilidad para la programación.

Refiriéndonos al programa en sí, podemos observar en listado que está organizado en una serie de subrutinas de utilidad que son llamadas desde el programa principal.

100-130 Subrutina de presentación y final.
200-398 Subrutina que dibuja en pantalla el trazado de la habitación, los marcadores y la clave de desactivación.
400-420 Subrutina que busca una posición al azar y la coloca en la variable X.

450-495 Subrutina que se encarga de generar la clave de desactivación de cada habitación.
500-535 Subrutina que se encarga de gestionar la muerte del individuo, comprueba si quedan vidas y en caso negativo finaliza la partida y propone la revancha.
600-615 Subrutina que se encarga de añadir los BONUS necesarios al marcador cuando se ha completado una pantalla.
700-720 Subrutina que se encarga de elevar el nivel de juego o las vidas si procede.
1000 Borra la pantalla y coloca los colores de fondo.
1010-1030 Inicializa la partida y pregunta el nivel inicial de juego.
1060-1070 Inicializa variables y pinta la habitación.
1080-1110 Coloca al azar las válvulas a lo largo de la habitación.
1120-1140 Se encargan de calcular la posición del vehículo según las teclas que se hayan pulsado.
1143 Comprueba si el vehículo ha cerrado una válvula.
1145-1200 Disminuye el oxígeno y actualiza los marcadores.
1210-1240 Comprueban si la válvula sigue la cadencia de la clave y genera la puntuación y la presentación en pantalla correspondiente.

Variables principales

PS y CO son utilizadas para la presentación
NI almacena el nivel de juego
VI almacena las vidas
SC almacena la puntuación
OX almacena el oxígeno
P1, P2, P3 utilizadas para los "peeks"
X, I, K utilizadas en labores menores
CL$ almacena la clave
CN es utilizado para determinar si la cadencia de desarticulación es correcta
SD antigua posición vehículo
OSD actual posición vehículo
CL utilizada en la elaboración de la clave.

```
2 REM SABOTAJE NUCLEAR
3 REM
4 REM AUTOR : FRANCISCO
5 REM         SARRIAS PEDEMONTE
6 REM
7 REM PARA EL CBM=64
8 REM
10 GOTO 1000
100 FOR I=1 TO 100
110 PS=INT(RND(1)*975):CO=INT(RND(1)
*255)
120 POKE1024+PS,81:POKE 55296+PS,CO
130 NEXT I:RETURN
200 PRINT"[CLR]****NIVEL***VIDAS***P
UNTOS***OXIGENO****";
210 PRINT"*****";NI;"*****";VI;"****
*";SC;"******";OX;
215 PRINT
300 PRINT"[WHT][40COMM+]";
305 PRINT"[COMM+][8SPC][COMM+][14SPC]
[COMM+][14SPC][COMM+]";
310 PRINT"[COMM+][8SPC][COMM+][14SPC]
[COMM+][14SPC][COMM+]";
315 PRINT"[COMM+][8SPC][COMM+][14SPC]
[COMM+][14SPC][COMM+]";
320 PRINT"[5COMM+][4SPC][COMM+][29SPC]
[COMM+]";
325 PRINT"[COMM+][38SPC][COMM+]";
330 PRINT"[COMM+][38SPC][COMM+]";
335 PRINT"[COMM+][8SPC][COMM+][29SPC]
[COMM+]";
340 PRINT"[COMM+][8SPC][COMM+][14SPC]
[COMM+][14SPC][COMM+]";
345 PRINT"[COMM+][8SPC][COMM+][14SPC]
[COMM+][14SPC][COMM+]";
350 PRINT"[COMM+][5SPC][10COMM+][8SPC]
[COMM+][7SPC][2COMM+][4SPC][2COMM+]"
;
355 PRINT"[COMM+][23SPC][COMM+][7SPC]
[COMM+][6SPC][COMM+]";
360 PRINT"[COMM+][22SPC][6COMM+][3SPC]
[COMM+][6SPC][COMM+]";
365 PRINT"[COMM+][17SPC][5SPC][COMM+]
[8SPC][COMM+][6SPC][COMM+]";
370 PRINT"[COMM+][22SPC][COMM+][5SPC]
[3SPC][COMM+][6SPC][COMM+]";
375 PRINT"[COMM+][4SPC][6COMM+][12SPC]
[COMM+][8SPC][COMM+][6SPC][COMM+]";
380 PRINT"[COMM+][9SPC][COMM+][12SPC]
[COMM+][8SPC][COMM+][6SPC][COMM+]";
385 PRINT"[COMM+][9SPC][COMM+][8SPC]
[5COMM+][8SPC][COMM+][6SPC][COMM+]";

390 PRINT"[COMM+][9SPC][COMM+][21SPC]
[COMM+][6SPC][COMM+]";
395 PRINT"[40COMM+]";
398 PRINT"[RVSON][YEL][4SPC]CLAVE[SPC]
:";CL$;"[10SPC]":RETURN
400 X=INT(RND(1)*800)+120
410 IF PEEK(1024+X)<>32 THEN 400
420 RETURN
450 CL$="":FOR I=1 TO 9
455 CL(I)=INT(RND(1)*10)
460 FOR K=0 TO I-1
465 IF CL(I)=CL(K) THEN 455
470 NEXT K
475 CL$(I)=STR$(CL(I))
480 CL$=CL$+CL$(I)
485 NEXT I
```

```
490 CN=0:REM CONTADOR NUMERO
495 RETURN
500 VI=VI-1:IFVI>0GOTO535
505 GOSUB100
515 PRINT"[CLR][8CRSRD][6CRSRR]"
520 INPUT"DESEAS[SPC]OTRA[SPC]PARTID
A[SPC]S/N";PA$
525 IFPA$="N"THEN END
530 RUN
535 GOTO 1065
600 FORI=OXTO 0 STEP-100/NI
605 SC=SC+NI*100:GOSUB200
608 GOSUB700
610 NEXT I
615 GOTO1065
700 IFSC/5000>INT(SC/5000)ANDSC/5000
<INT(SC/5000)*1.05THENNI=NI+1
710 IFSC/15000>INT(SC/15000)ANDSC/15
000<INT(SC/15000)*1.02THENVI=VI+1
720 RETURN
1000 PRINT"[CLR]": POKE53280,0:POKE5
3281,0
1010 GOSUB100
1020 PRINT"[CLR][10CRSRD][4SPC]";
1030 INPUT"[SPC]INTRODUCE[SPC]NIVEL[SPC]
1-5";NI:PRINT"[CLR]"
1060 VI=3:SC=0:DIMCL$(9),CL(9)
```

```
1065 OX=999:GOSUB450:GOSUB200
1070 GOSUB 400:SD=X:OSD=SD
1080 FOR I=49 TO 57
1090 GOSUB 400:REM X ALEATORIA
1100 POKE1024+X,I:POKE55296+X,8
1110 NEXT I
1120 P1=PEEK(197):P2=1024+SD
1130 SD=SD+1*(P1=26 AND PEEK(P2-1)<>
102)-1*(P1=34 AND PEEK(P2+1)<>102)
1140 SD=SD+40*(P1=25 AND PEEK(P2-40)
<>102)-40*(P1=39 AND PEEK(P2+40)<>10
2)
1143 P3=PEEK(1024+SD):IF P3>48 AND P
3<58THEN 1210
1145 OX=OX-NI:IFOX<0THENGOSUB500
1150 POKE1024+OSD,32:POKE1024+SD,81:
OSD=SD
1190 PRINT"[HOM][CRSRD][5SPC]";NI;"[5SPC]
";VI;"[5SPC]";SC;"[6SPC]";OX;
1200 GOTO 1120
1210 CN=CN+2:IFCN=18THENGOSUB600
1220 IFP3-48<>VAL(MID$(CL$,CN,1))THE
N GOSUB500
1230 SC=SC+INT(RND(1)*25)*10
1235 GOSUB700
1240 POKE56186+CN,2:GOTO 1145
```



Laberinto

VIC-20

Carlos Rojas López
Gustavo Gallardo, 7. 4ºD
Sevilla-13

Esta es la segunda vez que escribo a la revista, enviando un programa. Sin embargo sería tan malo que ni lo han publicado. Espero que esta vez no ocurra lo mismo.

Les mando un sencillo programa de juego; tal vez algunos opinen que es una versión del popular Pack-Man, pero yo opino que es muy original.

El juego consiste en mover al packman comiéndose los puntitos con el joystick; verás los puntos en la parte superior izquierda, pero debes tener cuidado con los monstruos que te acechan. En la parte superior derecha hay un cronómetro que te indica los segundos que faltan para que los monstruos aparezcan. Cada vez que salgan tardarán en aparecer un segundo menos, hasta los 15 segundos, en el que retornará a los 30 iniciales.

De vez en cuando saldrá una cereza que vale 500 puntos. Cuando consigas 2.200 puntos, puedes comerte el signo gráfico de arriba a la izquierda.

Cada 7.500 puntos tendrás un packman más —tienes 3 al principio—. El juego termina cuando agotes tus tres packman.

NOTA: los monstruos te comen cuando coinciden con tu packman o con las bombas que deja tras una aparición completa de todos los monstruos.

Como podéis comprobar, el juego es muy sencillo y está dividido en dos partes, o sea, grabado en dos partes:

1) Esta parte crea los gráficos de alta resolución que definen las figuras o muñecos que aparecerán durante el juego. Consta

GLOSARIO

A.—Amperio, analógico, asíncrono, sumador (adder).

ABEND.—(ABnormal END of task), es la terminación de una función o programa antes de su tiempo habitual, suele ser debido a un error producido durante la ejecución que no ha podido ser detectado.

Abort.—Es la terminación de una rutina dando un mensaje de error detectado por el sistema.

Absolute error.—(Error absoluto), es el rsultado de restar el valor obtenido o medido del valor teórico esperado.

A bus.—(Address bus), es el bus de direcciones de nuestro ordenador.

AC.—Corriente alterna (alternating current) o acoplador acústico (acoustic coupler).

ACC.—Acumulador.

Access.—(Acceso), es la manera en que nos referimos a los archivos o bloques de datos (directo, inmediato, serie, etc.).

Access time.—(Tiempo de acceso), es el intervalo de tiempo desde que la CPU solicita un dato hasta que lo lee (depende de la velocidad de la memoria o periféricos)

ACIA.—(Asynchronous communications interface adapter), es el dispositivo o chip que nos permite comunicarnos con un bus de comunicaciones serie asincrono (por ejemplo con un modem o impresora rs-232).

ACK.—(Acknowledgment), es la señal de reconocimiento, sirve para que un dispositivo indique al otro que ha recibido los datos que le envía.

Acoplador acústico.—(AC), es el dispositivo que permite convertir las señales enviadas por un modem a todos audibles para que estos sean transmitidos por el micrófono del teléfono a la línea telefónica y viceversa.

REPRODUCCION DE LISTADOS IMPRESOS DEFECTUOSAMENTE EN EL COMMODORE WORLD DE SEPTIEMBRE

Es la "Hora" de aprender, pág. 15 C-64

```
0 REM PARA EL RELOJ-C64
1 L$="132714091529168917671763176016
781518139813201323"
2 LH$="13291449152917691762176317581
638151813181320 1323"
3 IH$="+3978+3978-0145-4177-4177-406
6-3978-3978+0145+4177+4177+4066"
4 IP$="+3978+3978-0164-4177-4177-406
6-3978-3978+0167+4177+4177+4066"
5 B$="1243124913311531177118491843I8
381756151613161238":S2=54285
6 CO=54272:V=54296:SC=53281:BO=53280
:W2=54283:A2=54284:H2=54280:L2=54279

8 FORX=54272TO54296:POKEX,0:NEXTX:PO
KEV,15:POKEA2,116:POKES2,128
10 POKESC,12:POKEBO,3
11 PRINT"[CLR][5CRSRR][COMM4][RVSON]
[SPC]DIAGRAMA[SPC]DE[SPC]HORA[SPC][RVSOFF]
":PRINT"[CRSRD]";SPC(8);"00":PRINT"[6SPC]
[5COMM@][SPC]5"
15 PRINT"[3SPC]55[SHIFTN][2SPC]12[SPC]
[SHIFTM]":PRINT"[4SPC][SHIFTN]11[3SPC]
1[SPC][SHIFTM][SPC]19":PRINT"[SPC]50
[SHIFTN][9SPC][SHIFTM]"
20 PRINT"[2SPC][COMMM]10[SPC]RELOJ[SPC]
2[SPC][COMMG]":PRINT"[2SPC][COMMM]";
SPC(11);"[COMMG]"
25 PRINT"45[COMMM]9[2SPC][SHIFTQ][SHIFT*]
[COMM+][4SPC]3[COMMG]15":PRINT"[2SPC]
[COMMM][5SPC][SHIFT-][5SPC][COMMG]"
30 PRINT"[2SPC][COMMM]8[4SPC][SHIFT-]
[3SPC]4[SPC][COMMG]":PRINT"[SPC]40[SHIFTM]
[4SPC][SHIFT-][4SPC][SHIFTN]20"
35 PRINT"[4SPC][SHIFTM][SPC]7[SPC][SHIFTQ]
[SPC]5[SPC][SHIFTN]":PRINT"[3SPC]35[SHIFTM]
[2SPC]6[2SPC][SHIFTN]25"
40 PRINT"[6SPC][5COMMT]":PRINTTAB(7)
"30"
45 PRINT"[CRSRD][4SPC]HORA=[SPC]9:30
"
50 PRINT"[2CRSRD][SPC]PULSA[SPC]'F7'
[SPC]PARA[SPC]COMENZAR[SPC],O[2SPC]P
ULSA":PRINT"[SPC]'I'[SPC]PARA[SPC]IN
STRUCCIONES."
55 GETS$:IFS$=""THEN55
56 IFS$="I"THEN58
57 GOTO200
58 PRINT"[CLR]";SPC(11);"[RVSON][SPC]
INSTRUCCIONES[SPC][RVSOFF]"
70 PRINT"[CRSRD][SPC]GANAS[SPC]ACERT
ANDO[SPC]3[SPC]VECES."
75 PRINT"[2CRSRD][RVSON][SPC]TECLAS[SPC]
DE[SPC]CONTROL[SPC][RVSOFF]":PRINT"[2CRSRD]
'F1'[SPC]MUEVE[SPC]LA[SPC]MANILLA[SPC]
CHICA."
80 PRINT"[CRSRD]'F3'[SPC]MUEVE[SPC]L
A[SPC]MANILLA[SPC]GRANDE."
85 PRINT"[CRSRD]'F7'[SPC]PARA[SPC]EL
[SPC]RELOJ."
87 PRINT"[3CRSRD][SPC](PULSA[SPC]UNA
[SPC]TECLA.)"
90 GETS$:IFS$=""THEN90
150 POKESC,11:POKEBO,7:PRINT"[CLR][WHT]
[CRSRD][RVSON]DIFICULTAD[SPC]DE[SPC]
LA[SPC]PRUEBA[RVSOFF]"
155 PRINT"[CRSRD](1)[SPC]HORAS":PRIN
T"[CRSRD](2)[SPC]HORAS[SPC]Y[SPC]1/2
-HORAS"
160 PRINT"[CRSRD](3)[SPC]INTERVALOS-
5[SPC]MIN."
165 PRINT"[6CRSRD][2SPC]PULSA[SPC]1,
2,O[2SPC]3":PRINT"[CRSRD][SPC]O[2CRSRD]
[2SPC]PULSA[SPC]'F'[SPC]PARA[SPC]FIN
."
170 GETD$:IFD$=""THEN170
173 IFD$="F"THENPOKEV,0:POKESC,6:POK
EBO,14:PRINT"[CLR][COMM7]":END
175 D=VAL(D$)
180 IFD>3ORD<1THEN170
200 PRINT"[CLR][RED]":R=0:W=0:POKESC
,8:POKEBO,2
210 PRINTTAB(12)"[15COMM@]":PRINTTAB
(10)"[RVSON][SHIFT£][RVSOFF][SHIFT£]
";SPC(15);"[SHIFTM]"
211 PRINTTAB(9)"[RVSON][SHIFT£][RVSOFF]
[SHIFT£][SPC]11[5SPC]12[6SPC]1[SHIFTM]
":PRINTTAB(8)"[RVSON][SHIFT£][RVSOFF]
[SHIFT£]";SPC(19);"[SHIFTM]"
212 PRINTTAB(8)"[RVSON][SPC][RVSOFF]
10";SPC(18);"2[SHIFTM]":PRINTTAB(8)"
[RVSON][SPC][RVSOFF]";SPC(22);"[COMMG]
"
213 PRINTTAB(8)"[RVSON][SPC][RVSOFF]
";SPC(22);"[COMMG]":PRINTTAB(8)"[RVSON]
[SPC][RVSOFF]";SPC(22);"[COMMG]"
214 PRINTTAB(8)"[RVSON][SPC][RVSOFF]
";SPC(22);"[COMMG]":PRINTTAB(8)"[RVSON]
[SPC][RVSOFF]";SPC(22);"[COMMG]"
215 PRINTTAB(8)"[RVSON][SPC][RVSOFF]
";SPC(9);"[RVSON][4SPC][RVSOFF]";SPC
(9);"[COMMG]"
216 PRINTTAB(8)"[RVSON][SPC][RVSOFF]
[SPC]9";SPC(7);"[RVSON][4SPC][RVSOFF]
";SPC(7);"3[SPC][COMMG]"
217 PRINTTAB(8)"[RVSON][SPC][RVSOFF]
";SPC(9);"[RVSON][4SPC][RVSOFF]";SPC
(9);"[COMMG]"
218 PRINTTAB(8)"[RVSON][SPC][RVSOFF]
";SPC(9);"[RVSON][4SPC][RVSOFF]";SPC
(9);"[COMMG]"
219 PRINTTAB(8)"[RVSON][SPC][RVSOFF]
";SPC(22);"[COMMG]":PRINTTAB(8)"[RVSON]
[SPC][RVSOFF]";SPC(22);"[COMMG]"
220 PRINTTAB(8)"[RVSON][SPC][RVSOFF]
";SPC(22);"[COMMG]":PRINTTAB(8)"[RVSON]
[SPC][RVSOFF]";SPC(22);"[COMMG]"
221 PRINTTAB(8)"[RVSON][SPC][RVSOFF]
";SPC(22);"[COMMG]":PRINTTAB(8)"[COMM*]
[RVSON][COMM*][RVSOFF]8";SPC(18);"4[SHIFTN]
"
222 PRINTTAB(9)"[COMM*][RVSON][COMM*]
[RVSOFF]";SPC(18);"[SHIFTN]":PRINTTA
B(10)"[COMM*][RVSON][COMM*][RVSOFF]7
";SPC(6);"6";SPC(7);"5[SHIFTN]"
223 PRINTTAB(11)"[COMM*][RVSON][COMM*]
[RVSOFF][14COMM@][SHIFTN]"
```

```
240 HR=INT(RND(1)*12+1):Q=0
242 IFD=3THENMN=INT(RND(1)*12):GOTO2
48
243 IFD=2THENMN=6*INT(RND(1)*2):GOTO
248
247 MN=0
248 IFMN=0ORMN=1THENPRINT"[HOM]";HR;
":0";5*MN:GOTO250
249 PRINT"[HOM]";HR;":";5*MN
250 H=1 :RM=0
251 IFRM>2THENPL=VAL(MID$(LH$,(H-1)*
4+1,4)):GOTO280
261 IFRM>11THENH=H+1:RM=0
262 IFH>12THEN250
270 PL=VAL(MID$(L$,(H-1)*4+1,4))
280 PB=VAL(MID$(B$,4*RM+1,4))
281 PRINT"[HOM][14CRSRR][12SPC][10CRSRL]
";H;5*RM;
284 IFRM=0THENII=40:PP=66:GOTO290
285 II=VAL(MID$(IP$,(RM-1)*5+1,3))
286 PP=VAL(MID$(IP$,(RM-1)*5+4,2))
289 IF(RM>11)OR(H>12)THEN261
290 POKEPB,160:POKEPB+CO,0
300 FORK=1TO5:POKEPB+K*II,PP:POKEPB+
K*II+CO,0:NEXTK
305 POKEH2,30:POKEL2,141:POKEW2,17:F
ORK=1TO10:NEXTK:POKEW2,0
310 IFRM>2THENI=VAL(MID$(IH$,(H-1)*5
+1,3)):P=VAL(MID$(IH$,(H-1)*5+4,2)):
GOTO330
320 I=VAL(MID$(IP$,(H-1)*5+1,3)):P=V
AL(MID$(IP$,(H-1)*5+4,2))
330 POKEPL,160:POKEPL+CO,6
340 FORK=1TO3:POKEPL+K*I,P:POKEPL+K*
I+CO,6:NEXTK
350 FORK=1TO600:NEXTK
353 IFQ2=1THEN650
355 GETS$:IFS$=""THEN379
357 IFS$="[F1]"THENRM=RM+12:GOTO379
358 IFS$="[F3]"THENRM=RM+2:GOTO379
359 IFHR=HANDMN=RMTHEN500
360 GOTO600
379 POKEH2,61:POKEL2,126:POKEW2,17:F
ORK=1TO10:NEXTK:POKEW2,0
380 POKEPB,32:FORK=1TO5:POKEPB+K*II,
32:NEXTK
381 POKEPL,32:FORK=1TO3:POKEPL+K*I,3
2:NEXTK
385 IFQ=1THEN240
386 IFQ2=1THEN251
390 RM=RM+1
391 IFRM>11THENH=H+1:RM=0
392 IFH>12THEN250
400 GOTO251
500 FORK=1TO2:PRINT"[HOM]"SPC(20);"S
I"
501 FORX=1TO10:POKEH2,108:POKEL2,223
:POKEW2,17:FORJ=1TO10:NEXTJ
502 POKEW2,0:NEXTX
503 PRINT"[HOM]";SPC(20);"[4SPC]":FO
RX=1TO100:NEXTX:NEXTK
504 R=R+1:W=0:PRINT"[HOM]";SPC(37);R

505 IFR=3THENPRINT"[HOM][SPC][RVSON]
FANTASTICO[RVSOFF][10SPC]":GOTO507
506 PRINT"[HOM]PULSA[SPC]'F7'[SPC]PA
RA[SPC]EMPEZAR.":GOTO509
507 FORX=1TO75:POKEW2,33:POKEH2,X:PO
KEL2,100+2*X:FORJ=1TO10:NEXTJ:NEXTX
508 POKEW2,0:GOTO700
509 GETS$:IFS$=""THEN509
510 PRINT"[HOM][24SPC]":Q=1:GOTO380
600 POKEH2,8:POKEL2,23:POKEW2,33:FOR
X=1TO900:NEXTX:POKEW2,0
601 Q2=1:RM=MN:H=HR:W=W+1
603 GOTO380
650 PRINT"[HOM]";SPC(9);"CORREGIDO":
Q2=0
652 FORX=1TO2000:NEXTX
653 IFW>2THEN10
654 PRINT"[HOM]PULSA[SPC]'F7'[SPC]PA
RA[SPC]EMPEZAR.[2SPC]"
655 GETS$:IFS$=""THEN655
657 PRINT"[HOM][26SPC]"
660 Q=1:GOTO380
700 FORX=1TO1000:NEXTX
701 GOTO150
```

Es la "Hora" de aprender, pág. 18 VIC-20

```
0 REM PARA EL RELOJ VIC-20
1 L$="782678497915800380248020801779
947906784078197822"
2 LH$="782778717937802580238020801 67
97279067818782 07822"
3 IH$="+2178+2178-0164-2377-2377-226
6-2178-2178+0164+2377+2377+2266"
4 IP$="+2178+2178-0164-2377-2377-226
6-2178-2178+0167+2377+2377+2266"
5 B$="777877847807791780498070806480
598036790477947773"
6 CO=30720:V=36878:S1=36874:S2=36875
:S3=36876:SC=36879:POKEV,15
10 POKESC,174:PRINT"[CLR][3CRSRR][RVSON]
DIAGRAMA[SPC]DE[SPC]RELOJ[RVSOFF]":P
RINT"[CRSRD]";SPC(8);"00":PRINT"[6SPC]
[5COMM@]5"
15 PRINT"[3SPC]55[SHIFTN][2SPC]12[SPC]
[SHIFTM]":PRINT"[4SPC][SHIFTN]11[4SPC]
1[SHIFTM][SPC]10":PRINT"[SPC]50[SHIFTN]
[9SPC][SHIFTM]"
20 PRINT"[2SPC][COMMM]10[SPC]RELOJ[2SPC]
2[COMMG]":PRINT"[2SPC][COMMM]";SPC(1
1);"[COMMG]"
25 PRINT"45[COMMM]9[2SPC][SHIFTQ][SHIFT*]
[COMM+][5SPC][COMMG]15":PRINT"[2SPC]
[COMMM][5SPC][SHIFT-][5SPC][COMMG]"
30 PRINT"[2SPC][COMMM]8[4SPC][SHIFT-]
[4SPC]4[COMMG]":PRINT"[SPC]40[SHIFTM]
[4SPC][SHIFT-][4SPC][SHIFTN]20"
35 PRINT"[4SPC][SHIFTM]7[2SPC][SHIFTQ]
[2SPC]5[SHIFTN]":PRINT"[3SPC]35[SHIFTM]
[2SPC]6[2SPC][SHIFTN]25"
40 PRINT"[6SPC][5COMMT]":PRINTTAB(7)
"30"
45 PRINT"[CRSRD][4SPC]HORA=[SPC]9:30
"
50 PRINT"[CRSRD]PULSA[SPC]'F7'PARA[SPC]
EMPEZAR[CRSRD]O[SPC]'I',[SPC]INSTRUC
CIONES.";
55 GETS$:IFS$=""THEN55
56 IFS$="I"THEN58
57 GOTO200
58 PRINT"[CLR][4SPC][RVSON][RED]INST
RUCCIONES[RVSOFF][BLU]":PRINT"[CRSRD]
INTENTA[SPC]PARAR[SPC]EL[SPC]RELOJ"
65 PRINT"CUANDO[SPC]LAS[SPC]MANILLAS
[SPC]":PRINT"[CRSRD]INDIQUEN[SPC]LA[SPC]
HORA."
70 PRINT"[CRSRD]GANAS[SPC]CON[SPC]3[SPC]
ACIERTOS."
75 PRINT"[2CRSRD][RVSON]TECLAS[SPC]D
E[SPC]CONTROL[RVSOFF]":PRINT"[CRSRD]
'F1'[SPC]MUEVE[SPC]LA[SPC]MANILLA[6SPC]
CHICA"
80 PRINT"[CRSRD]'F3'[SPC]MUEVE[SPC]L
A[SPC]GRANDE"
85 PRINT"[CRSRD]'F7'[SPC]PARA[SPC]EL
[SPC]RELOJ"
87 PRINT"[2CRSRD](PULSA[SPC]UNA[SPC]
TECLA.)";
90 GETS$:IFS$=""THEN90
150 PRINT"[CLR][CRSRD][SPC][RVSON]DI
```

```
FICULTAD[SPC]DEL[SPC]JUEGO"
155 PRINT"[2CRSRD](1)[SPC]HORAS":PRI
NT"[CRSRD](2)[SPC]HORAS[SPC]Y[SPC]1/
2[SPC]HORAS"
160 PRINT"[CRSRD](3)[SPC]INTERVALOS[2SPC]
DE[2SPC]5[5SPC][CRSRD]MINUTOS."
165 PRINT"[5CRSRD]PULSA[SPC]1,2,O[SPC]
3.":PRINT"[2CRSRD]PULSA[SPC]'F'[SPC]
PARA[SPC]FIN."
170 GETD$:IFD$=""THEN170
173 IFD$="F"THENPOKEV,0:POKESC,27:PR
INT"[CLR]":END
175 D=VAL(D$)
180 IFD>3ORD<1THEN170
200 PRINT"[CLR]":R=0:W=0:POKESC,159
210 PRINT"[4CRSRR][14COMM@]"
211 PRINT"[2CRSRR][RVSON][SHIFT£][RVSOFF]
[SHIFT£]11[4CRSRR]12[5CRSRR]1[SHIFTM]
"
212 PRINT"[CRSRR][RVSON][SHIFT£][RVSOFF]
[SHIFT£]";SPC(16);"[SHIFTM]"
213 PRINT"[CRSRR][RVSON][SPC][RVSOFF]
10";SPC(15);"2[COMMG]"
214 PRINT"[CRSRR][RVSON][SPC][RVSOFF]
";SPC(18);"[COMMG]":PRINT"[CRSRR][RVSON]
[SPC][RVSOFF]";SPC(18);"[COMMG]":PRI
NT"[CRSRR][RVSON][SPC][RVSOFF]"SPC(1
8);"[COMMG]"
215 PRINT"[CRSRR][RVSON][SPC][RVSOFF]
";SPC(18);"[COMMG]":PRINT"[CRSRR][RVSON]
[SPC][RVSOFF]"SPC(18);"[COMMG]"
216 PRINT"[CRSRR][RVSON][SPC][RVSOFF]
9[7CRSRR][RVSON][2SPC][RVSOFF][7CRSRR]
3[COMMG]"
217 PRINT"[CRSRR][RVSON][SPC][RVSOFF]
[8CRSRR][RVSON][2SPC][RVSOFF][8CRSRR]
[COMMG]"
218 PRINT"[CRSRR][RVSON][SPC][RVSOFF]
";SPC(18);"[COMMG]":PRINT"[CRSRR][RVSON]
[SPC][RVSOFF]";SPC(18);"[COMMG]":PRI
NT"[CRSRR][RVSON][SPC][RVSOFF]";SPC(
18);"[COMMG]"
219 PRINT"[CRSRR][RVSON][SPC][RVSOFF]
";SPC(18);"[COMMG]":PRINT"[CRSRR][RVSON]
[SPC][RVSOFF]";SPC(18);"[COMMG]"
220 PRINT"[CRSRR][RVSON][SPC][RVSOFF]
8";SPC(16);"4[COMMG]":PRINT"[CRSRR][RVSON]
[SPC][RVSOFF]";SPC(18);"[COMMG]"
221 PRINT"[CRSRR][COMM*][RVSON][COMM*]
[RVSOFF]7[6CRSRR]6[6CRSRR]5[CRSRR][SHIFTN]
": PRINT"[3CRSRR][16COMMT]"
240 HR=INT(RND(1)*12+1):Q=0
242 IFD=3THENMN=INT(RND(1)*12):GOTO2
48
243 IFD=2THENMN=6*INT(RND(1)*2):GOTO
248
247 MN=0
248 IFMN=0ORMN=1THENPRINT"[HOM]";HR;
":[SPC]";RIGHT$("0"+RIGHT$(STR$(5*MN
),1),2):GOTO250
249 PRINT"[HOM]";HR;":";5*MN:
250 H=1 :RM=0
251 IFRM>2THENPL=VAL(MID$(LH$,(H-1)*
4+1,4)):GOTO280
270 PL=VAL(MID$(L$,(H-1)*4+1,4))
280 PB=VAL(MID$(B$,4*RM+1,4))
284 IFRM=0THENII=22:PP=66:GOTO290
285 II=VAL(MID$(IP$,(RM-1)*5+1,3))
286 PP=VAL(MID$(IP$,(RM-1)*5+4,2))
290 POKEPB,160:POKEPB+CO,0
300 FORK=1TO5:POKEPB+K*II,PP:POKEPB+
K*II+CO,0:NEXTK
305 POKES1,240:FORK=1TO10:NEXTK:POKE
S1,0
310 IFRM>2THENI=VAL(MID$(IH$,(H-1)*5
+1,3)):P=VAL(MID$(IH$,(H-1)*5+4,2)):
GOTO330
320 I=VAL(MID$(IP$,(H-1)*5+1,3)):P=V
AL(MID$(IP$,(H-1)*5+4,2))
330 POKEPL,160:POKEPL+CO,6
340 FORK=1TO3:POKEPL+K*I,P:POKEPL+K*
I+CO,6:NEXTK
350 FORK=1TO600:NEXTK
353 IFQ2=1THEN650
355 GETS$:IFS$=""THEN379
357 IFS$="[F1]"THENRM=RM+12:GOTO379
358 IFS$="[F3]"THENRM=RM+2:GOTO379
359 IFHR=HANDMN=RMTHEN500
360 GOTO600
379 POKES2,240:FORK=1TO10:NEXTK:POKE
S2,0
380 POKEPB,32:FORK=1TO5:POKEPB+K*II,
32:NEXTK
381 POKEPL,32:FORK=1TO3:POKEPL+K*I,3
2:NEXTK
385 IFQ=1THEN240
386 IFQ2=1THEN251
390 RM=RM+1
391 IFRM>11THENH=H+1:RM=0
392 IFH>12THEN250
400 GOTO251
500 FORK=1TO2:PRINT"[HOM]"SPC(14);"S
I[4SPC]";R+1;
501 FORJ=1TO15:FORX=1TO10:POKES3,230
:NEXTX:POKES3,0:NEXTJ
502 PRINT"[HOM]";SPC(14);"[4SPC]":PO
KES3,0:FORX=1TO200:NEXTX:NEXTK
503 R=R+1:W=0:PRINT"[HOM]";SPC(20);R

504 IFR=3THENPRINT"[HOM]FANTASTICO![7SPC]
":FORX=1TO50:POKES3,190+X:NEXTX:POKE
S3,0:GOTO700
505 PRINT"[HOM]PULSA[SPC]F7[SPC]PARA
[SPC]EMPEZAR.";
506 GETS$:IFS$=""THEN506
510 PRINT"[HOM][22SPC]";:Q=1:GOTO380

600 POKES1,190:FORX=1TO400:NEXTX:POK
ES1,0:Q2=1:RM=MN:H=HR:W=W+1
603 GOTO380
650 PRINT"[HOM]";SPC(9);"CORREGIDO":
Q2=0
652 FORX=1TO2000:NEXTX
653 IFW>2THEN10
654 PRINT"[HOM]PULSA[SPC]F7[SPC]PARA
[SPC]EMPEZAR.";
655 GETS$:IFS$=""THEN655
657 PRINT"[HOM][22SPC]";
660 Q=1:GOTO380
700 FORX=1TO1000:NEXTX
701 GOTO150
```

Bingo, pág. 39

```
1 POKE53280,5:POKE53281,5
3 A=1305:B=1311:C=54272:W=1032:E=134
8:F=1828:G=1960:DIMV(90)
4 PRINT"[CLR][19CRSRD][11CRSRR][BLK]
ESPERE[SPC]POR[SPC]FAVOR":FORT=1TO30
00:NEXTT
5 GOSUB880
10 GOSUB680
20 LL=0:BB=0:P=0:PRINT"[CLR][BLK]"
30 FORI=W-8TOW:POKEI,121:POKEI+320,1
20:NEXTI
40 FORT=W+32TOA-1STEP40:POKET,117:PO
KET+8,118:NEXTT
50 FORI=E-1TOF-41STEP40:POKEI,118:PO
KEI+2,117:NEXTI
60 FORT=F+39TOF+41:POKET,120:NEXTT
70 FORI=1TO9:FORT=1TO10:W=W+3:POKEW,
```

```
102:POKEW+1,102:NEXTT:W=W+50:NEXTI
80 FORI=1TO90:V(I)=I:NEXTI:MAX=90:TT
=1
85 PRINT"[22CRSRD]NUMERO[SPC]ANTERIO
R"
87 PRINTTAB(19)"[2CRSRU]PULSAR[SPC][RVSON]
L[RVSOFF][SPC]PARA[SPC]LINEA"
88 PRINTTAB(19)"PULSAR[SPC][RVSON]B[RVSOFF]
[SPC]PARA[SPC]BINGO[4CRSRU]"
90 FORI=ATOB:POKEI+C,6:POKEI,81:NEXT
I
95 POKEC+24,15:POKEC+6,128:POKEC+5,1
28
100 IFTT>90THEN630
110 NU=INT(RND(TI)*MAX)+1:W=1032:GOS
UB400
130 FORK=1TO2
140 X1=INT(RND(TI)*(B-A+1))+A
150 X2=INT(RND(TI+1)*(B-A+1))+A
160 IFX2=X1THEN150
170 X3=INT(RND(TI+2)*(B-A+1))+A
180 IFX3=X1ORX3=X2THEN170
190 L=-1:GOSUB300
200 L=1:X1=D-40:GOSUB300:GOSUB800:IF
P=1THENK=2
210 NEXTK
215 IFP=1THEN630
217 POKEC+4,33
220 FORI=A+3TO1788STEP40:POKEI+C,5:P
OKEI,32:POKEI+40+C,6:POKEI+40,81
225 POKEC+1,I/50:POKEC,I/200AND255:N
EXTI:POKEC+4,0
230 GOSUB800:IFP=1THEN630
240 POKEF+C,7:POKEF,FI+48:POKEF+1+C,
7:POKEF+1,CO+48:FORT=1TO300:NEXTT
250 POKEPO+C,1:POKEPO,FI+48:POKEPO+1
+C,1:POKEPO+1,CO+48
260 POKEG+C,1:POKEG,FI+48:POKEG+1+C,
1:POKEG+1,CO+48
270 GOSUB800:IFP=1THEN630
280 POKEF+C,5:POKEF,32:POKEF+1+C,5:P
OKEF+1,32:POKEE+C,0:POKEE,120
283 POKEA+3+C,6:POKEA+3,81
285 GOSUB800:IFP=1THEN630
290 GOSUB440:GOTO100
300 POKEC+4,17:I2=0:I3=0
310 FORI=1TO5
320 IFI>2THENI2=40
330 IFI>3THENI3=40
340 D=X1+40*L*(I-1)
350 POKED+C,5:POKED,32:POKEX2+C,5:PO
KEX2,32:POKEX3+C,5:POKEX3,32
360 X2=X2+I2*L:X3=X3+I3*L
365 NT=50+L*(12*I-42):POKEC+1,NT:POK
EC,NT/2AND255
370 POKEX2+C,6:POKEX2,81:POKEX3+C,6:
POKEX3,81:POKED+C+40*L,6:POKED+40*L,
81
380 NEXTI:POKEC+4,0:RETURN
400 FI=INT(V(NU)/10):CO=V(NU)-FI*10:
PO=W+80*FI
410 IFCO<>0THENPO=PO+CO*3:GOTO430
420 PO=PO-50
430 RETURN
440 IFNU=MAXTHEN460
450 FORT=NUTOMAX-1:V(T)=V(T+1):NEXTT

460 MAX=MAX-1:TT=TT+1
470 RETURN
475 GOSUB1130
480 IFLL=1ANDS$="L"THENPRINT"LINEA[SPC]
YA[SPC]CANTADA":GOTO550
490 IFBB=1ANDS$="B"THENPRINT"BINGO[SPC]
YA[SPC]CANTADO":GOTO550
500 IFS$="L"THENLL=1:PRINT"LINEA[SPC]
CORRECTA[SPC](S/N)?":GOSUB560:GOTO57
0
510 IFS$="B"THENBB=1:PRINT"BINGO[SPC]
CORRECTO[SPC](S/N)?":GOSUB560:GOTO59
0
520 PRINT"ERROR[SPC]AL[SPC]PULSAR":P
RINTTAB(17)"PULSE[SPC]DE[SPC]NUEVO[SPC]
[RVSON]L[RVSOFF][SPC]O[SPC][RVSON]B[RVSOFF]
"
525 GETS$:IFS$=""THEN525
530 GOSUB1170
540 GOTO 480
550 PRINTTAB(17)"CONTINUAMOS":FORK=1
TO3000:NEXTK:GOSUB1170:GOTO620
560 GETS$:IFS$=""THEN560
565 IFS$<>"S"ANDS$<>"N"THEN560
567 RETURN
570 IFS$="S"THENGOSUB1170:PRINT"LINE
A[SPC]CORRECTA":PRINTTAB(17)"GANA"GL
"PTS":GOTO550
580 IFS$="N"THENLL=0:GOSUB1170:PRINT
"LINEA[SPC]NO[SPC]CORRECTA":GOTO550
590 IFS$="N"THENBB=0:GOSUB1170:PRINT
"BINGO[SPC]NO[SPC]CORRECTO":GOTO550
600 IFS$="S"THENGOSUB1170:PRINT"BING
O[SPC]CORRECTO":PRINTTAB(17)"GANA"GB
"PTS":P=1
610 PRINTTAB(17)"FELICIDADES":RESTOR
E:GOSUB820:FORT=1TO400:NEXTT
620 RETURN
630 PRINT"[CLR][CRSRD]QUIERES[SPC]JU
GAR[SPC]OTRA[SPC]VEZ[SPC](S/N)?"
640 GETS$:IFS$=""THEN640
650 IFS$="S"THENPRINT"[CRSRD]ES[SPC]
LO[SPC]MEJOR[SPC]QUE[SPC]PUEDES[SPC]
HACER":FORT=1TO3000:NEXTT:GOTO10
660 IFS$<>"N"THEN630
670 PRINT"[CRSRD]TU[SPC]TE[SPC]LO[SPC]
PIERDES,[SPC]ADIOS":END
680 PRINT"[CLR]"TAB(18)"[BLK][RVSON]
BINGO[RVSOFF]":PRINT"[2CRSRD]":INPUT
"CARTONES[SPC]VENDIDOS";AA:PRINT
690 INPUT"PRECIO[SPC]DE[SPC]CADA[SPC]
CARTON";PP:PRINT
700 INPUT"%[SPC]PARA[SPC]LINEA";SS:P
RINT
710 INPUT"%[SPC]PARA[SPC]BINGO";CC:P
RINT
720 IFSS+CC>100THEN700
730 IFSS+CC<100THENGL=INT(AA*PP*SS/1
00):GB=INT(AA*PP*CC/100):GOTO750
740 GL=INT(AA*PP*SS/100):GB=AA*PP-GL

750 PRINT"[2CRSRD]PREMIO[SPC]A[SPC]L
A[SPC]LINEA"GL"PTS"
760 PRINT"[2CRSRD]PREMIO[2SPC]AL[2SPC]
BINGO"GB"PTS"
770 PRINT"[2CRSRD]PULSE[SPC]UNA[SPC]
TECLA[SPC]PARA[SPC]EMPEZAR"
780 GETS$:IFS$=""THEN780
790 RETURN
800 GETS$:IFS$<>""THENGOSUB475
810 RETURN
820 POKEC+24,15:POKEC+6,0:POKEC+5,12
8:POKEC+3,99:Z=3000
822 READM1:IFM1=0THENPOKEC+4,0:RETUR
N
824 R=INT(M1/10):Q=(M1-10*R)*Z
826 POKEC+1,Q/256:POKEC,QAND255
828 POKEC+4,0:POKEC+4,65:FORI=1TO40*
(R+1):NEXTI
830 GOTO822
832 DATA26,6,25,5,2,2,2,5,6,7,26,6,2
5,5,2,2,2,64,2,2,5,4,4,5,2,2,4,5,6,7

834 DATA15,6,24,5,2,2,2,5,6,7,26,6,2
5,5,2,2,2,94,0
880 PRINT"[CLR]":H=1027:HH=1427
890 FORI=HTOHHSTEP40:POKEI,42:POKEI+
8,42:POKEI+11,42:POKEI+16,42:NEXTI
900 FORI=H+59TOHH-21STEP40:POKEI,42:
POKEI+8,42:POKEI+13,42:NEXTI
910 FORI=H+1TOH+3:POKEI,42:POKEI+200
,42:POKEI+400,42:NEXTI
920 POKEH+4,42:POKEH+164,42:POKEH+24
```

```
4,42:POKEHH+4,42
930 FORI=H+45TOH+125STEP40:POKEI,42:
POKEI+240,42:NEXTI
940 FORI=H+94TOH+334STEP40:POKEI,42:
NEXTI
950 FORI=H+20TOH+23:POKEI,42:POKEI+8
,42:POKEI+400,42:POKEI+408,42:NEXTI
960 POKEH+12,42:POKEH+53,42:POKEH+37
5,42
970 FORI=H+302TOH+304:POKEI,42:NEXTI

980 POKEH+64,42:POKEH+344,42:POKEH+3
84,42
990 PRINT"[16CRSRD][6CRSRR]POR[SPC]R
AFAEL[SPC]MUNYOZ...":FORI=1TO8000:NE
XTI
1000 RETURN
1130 POKE781,18:POKE782,17:POKE783,0
:SYS65520
1140 RETURN
1150 PRINT"[22SPC]":PRINTTAB(17)"[22SPC]
"
1155 PRINTTAB(17)"[22SPC]"
1160 RETURN
1170 GOSUB1130
1180 GOSUB1150
1190 GOSUB1130
2000 RETURN
```

Halterofilia, pág. 41

```
10 POKE52,24:POKE56,24:CLR
20 POKE36869,254
22 FORI=6144TO6668:POKEI,PEEK(26624+
I):NEXTI
23 PRINT"[CLR][4CRSRD][10CRSRR][RVSON]
@A"
25 PRINT"[10CRSRR][RVSON]VW"
26 GOSUB3000
40 PRINT"[3CRSRD][3CRSRR]**HALTEROFI
LIA**"
41 PRINT"[4CRSRD][6CRSRR](19/V/84)"
42 FORT=1TO4000:NEXTT:POKE36879,11
43 PRINT"[CLR]":POKE36869,240
45 PRINT"CUANDO[SPC]OIGAS[SPC]EL[SPC]
PRIMER":PRINT"PITIDO,PULSA[SPC]LA[SPC]
TECLA"
46 PRINT"[CRSRD][3CRSRR]/CRSR(DERECH
A)/":PRINT:L=25
47 PRINT"EL[SPC]MAYOR[SPC]NUMERO[SPC]
DE":PRINT"[CRSRD]VECES[SPC]POSIBLE":
PRINT"[CRSRD]HASTA[SPC]QUE"
48 PRINT:PRINT"SUENE[SPC]EL[SPC]2[SPC]
PITIDO."
49 PRINT"[2CRSRD][3CRSRR][RVSON]PULS
A[SPC]UNA[SPC]TECLA[RVSOFF]"
50 GETX$:IFX$=""THEN50
58 POKE36869,240:POKE36879,145
59 PRINT"[CLR]":PRINT"[3CRSRD][CRSRR]
INTENTA[SPC]LEVANTAR";L:PRINT"[6CRSRR]
KILOS"
60 PRINT"[7CRSRD][6CRSRR]ATENCION!!"

65 FORT=1TO4000:NEXTT
67 GETS$:IFS$<>""THEN67
70 POKE36878,15:POKE36876,215:PRINT"
[3CRSRD][6CRSRR]!!!YA!!!"
75 FORT=1TO500:NEXTT:POKE36878,0
77 H=10:TI$="000000"
78 GETX$:IFX$="[CRSRR]"THENP=P+1
79 IFTI/60>HTHEN82
80 GOTO78
82 PRINT"[2CRSRD][5CRSRR]!!ALTO!!":P
OKE36878,15:POKE36876,215:POKE36879,
28
83 FORT=1TO500:NEXTT:POKE36878,0
84 IFL>PTHEN1000
85 POKE36878,15:FORL=130TO254:POKE36
876,L
86 FORM=1TO40:NEXTM:NEXTL:POKE36879,
143
87 PRINT"[CLR]":PRINT"[8CRSRD]TE[SPC]
HA[SPC]COSTADO[SPC]PERO..."
100 POKE52,24:POKE56,24:CLR:POKE3686
9,254
110 FORI=6144TO6668:POKEI,PEEK(26624
+I):NEXTI
115 PRINT"[CLR][4CRSRD][10CRSRR][RVSON]
@A"
117 PRINT"[10CRSRR][RVSON]VW"
120 GOSUB5000
140 PRINT"[3CRSRD][CRSRR]!!LO[SPC]CO
NSEGUISTE!!"
149 FORG=1TO4000:NEXTG:POKE36869,240
:POKE36879,63
150 PRINT"[CLR]":PRINT"[3CRSRD]CUANT
OS[SPC]KILOS[SPC]QUIERES":INPUT"[CRSRD]
AHORA";L
152 POKE36879,57:GOTO58
1000 PRINT"[CLR]":PRINT"[3CRSRD][4CRSRR]
LO[SPC]SIENTO,[SPC]SOLO"
1010 PRINT"[CRSRD]TENIAS[SPC]FUERZA[SPC]
PARA":PRINT"[CRSRD]LEVANTAR";P:PRINT
"KILOS"
1030 INPUT"OTRO[SPC]JUEGO(S/N)";Y$
1040 IFY$="S"THENGOTO42
1050 STOP
3000 FORI=7168TO7183:READX:POKEI,X:N
EXTI
3030 FORI=7344TO7359:READX:POKEI,X:N
EXTI
3060 RETURN
5000 FORI=7168TO7183:READX:POKEI,X:N
EXTI
5010 DATA48,51,63,59,57,15,15,7
5020 DATA12,204,252,220,156,240,240,
224
5030 FORI=7344TO7359:READX:POKEI,X:N
EXTI
5040 DATA7,7,7,7,3,3,3,7
5050 DATA224,224,224,224,192,192,192
,224
5060 RETURN
```

El Pistolero, pág. 44

```
2 GOSUB320
4 DATA60,66,36,24,31,24,36,66
5 DATA60,66,36,24,248,24,36,66
6 DATA129,165,165,165,255,24,24,24
7 DATA24,60,102,102,231,195,195,195
8 DATA195,195,255,255,66,66,66,66
9 DATA0,0,0,24,24,0,0,0
10 DATA24,24,24,24,24,24,60,60
11 POKE52,28:POKE56,28:CLR:FORI=7168
TO7168+7*8-1:READX:POKEI,X:NEXT:PRIN
T"[CLR]"CHR$(8)
13 POKE36869,255:TI$="000000"
15 FORI=1TO12:C=INT(RND(1)*18)+INT(R
ND(1)*21)*22:POKE7726+C,2:POKE38446+
C,0:NEXT
20 J1=8164:J2=8185:B1=6:B2=6:CP=3072
0:O=160:S=22:C1=0:C2=0:FORI=1TO5:C(I
)=7822+66*I:NEXT
30 PRINT"[HOM][RVSON]MUERTOS"M1"[HOM]
[12CRSRR]MUERTOS"M2
31 PRINT"[HOM][RVSON][CRSRD][9CRSRR]
```

▶

```
"VAL(MID$(TI$,3,2))*60+VAL(RIGHT$(TI
$,2)):GOSUB140
35 POKEJ1,0:POKEJ2,1:POKEJ1+CP,0:POK
EJ2+CP,0:POKEJ1+S,0:POKEJ1-S,0:POKEJ
2+S,0:POKEJ2-S,0
36 FORI=1TO5:POKEC(I),4:POKEC(I)-S,3
:POKECP+C(I),0:POKECP+C(I)-S,0:POKEC
(I)+S,0
37 C(I)=C(I)-22:IFC(I)=7734THENPOKEC
(I),0:POKEC(I)+22,0:C(I)=8174
38 NEXTI
39 K1=PEEK(653):K2=PEEK(197)
40 IFK1=1ANDC1=0THENC1=1:IFB1<0THENB
1=0
41 IFK2=31ANDC2=0THENC2=1:IFB2<0THEN
B2=0
50 IFK1=2THENJ1=J1+22:IFJ1=8186THENJ
1=8164
55 IFK1=4THENJ1=J1-22:IFJ1=7702THENJ
1=7724
60 IFK2=23THENJ2=J2+22:IFJ2=8207THEN
J2=8185
65 IFK2=54THENJ2=J2-22:IFJ2=7723THEN
J2=7745
80 IFC1<>0ANDB1<>0THENGOSUB160
85 IFC2<>0ANDB2<>0THENGOSUB190
110 IFTI$>="000300"THEN300
120 IFB1=0ANDB2=0THENPRINT"[CLR]":GO
TO15
130 GOTO30
140 PRINT"[HOM][CRSRD]";:IFB1=0THENP
RINT"[RVSON][6SPC]":C1=0:GOTO150
145 FORI=1TOB1:PRINT"F";:NEXTI:FORI=
1TO6-B1:PRINT"[RVSON][SPC][RVSOFF]";
:NEXTI
150 PRINT"[HOM][CRSRD][16CRSRR]";:IF
B2=0THENPRINT"[RVSON][6SPC]":C2=0:RE
TURN
155 IF 6-B2=0THEN157
156 FORI=1TO6-B2:PRINT"[RVSON][SPC][RVSOFF]
";:NEXTI
157 FORI=1TOB2:PRINT"F";:NEXTI:RETUR
N
160 B1=B1-1:D1=J1+1:FORC1=1TO20:A1=D
1+C1
165 IFPEEK(A1)=1THENM1=M1+1
170 IFPEEK(A1)=2ORPEEK(A1)=3ORPEEK(A
1)=4THENPOKEA1-1,0:GOTO185
175 POKEA1,5:POKECP+A1,0:POKEA1-1,0
180 NEXTC1
185 C1=0:POKEA1,0:RETURN
190 B2=B2-1:D2=J2-1:FORC2=1TO20:A2=D
2-C2
195 IFPEEK(A2)=0THENM2=M2+1
200 IFPEEK(A2)=2ORPEEK(A2)=3ORPEEK(A
2)=4THENPOKEA2+1,0:GOTO215
205 POKEA2,5:POKECP+A2,0:POKEA2+1,0
210 NEXTC2
215 POKEA2,0:C2=0:RETURN
300 PRINT"[HOM][4CRSRD][RVSON][2CRSRR]
OTRA[SPC]PARTIDA?[SPC](S/N)"
301 GETA$:IFA$="S"THENRUN
302 IFA$<>"N"THEN301
310 PRINT"[CLR]"CHR$(9):POKE36869,24
0:END
320 POKE36869,242:POKE36879,25:PRINT
CHR$(8)"[CLR][RVSON][BLK][4SPC][SHIFTL]
[SHIFTO][SHIFTS][SPC][SHIFTR][SHIFTI]
[SHIFTS][SHIFTT][SHIFTO][SHIFTL][SHIFTE]
[SHIFTR][SHIFTO][SHIFTS][4SPC]"
330 PRINT"[SPC][SHIFTJ]UEGO[SPC]PARA
[SPC]DOS[SPC]JUG.":PRINT"[CRSRD][RVSON]
[SPC][SHIFTM][SHIFTI][SHIFTS][SHIFTI]
[SHIFTO][SHIFTN][SPC][RVSOFF]:[SPC][SHIFTC]
ARGATE[SPC]AL"SPC(12)"CONTRARIO[SPC]
SI[SPC]"
340 PRINTSPC(10)"PUEDES."
350 PRINT"[2CRSRD][SHIFTF]UNCION[2SPC]
[SHIFTJ]UG.[SPC]1[SPC][SHIFTJ]UG[SPC]
2"
360 PRINT"[CRSRD][SPC][SHIFTA]RRIBA[2SPC]
[RVSON][SPC]CTRL[SPC][RVSOFF][2SPC][RVSON]
[2SPC]↑[2SPC][RVSOFF]"
365 PRINT"[SPC][SHIFTA]BAJO[3SPC][RVSON]
COMM.[SPC][RVSOFF][2SPC][RVSON]CRSR←
[RVSOFF]"
370 PRINT"[SPC][SHIFTD]ISPARO[SPC][RVSON]
SHIFT[SPC][RVSOFF][2SPC][RVSON]CRSR↑
[RVSOFF]"
375 PRINT"[CRSRD][SHIFTP]ULSA[SPC][RVSON]
[SHIFTS][SHIFTH][SHIFTI][SHIFTF][SHIFTT]
,[SHIFTC][SHIFTT][SHIFTR][SHIFTL],[SHIFTC]
[SHIFTO][2SHIFTM]."
380 IFPEEK(653)<>7THEN380
385 RETURN
```

Master Mind, pág. 45

```
10 REM**MASTER MIND**
15 DIM D(14)
20 POKE53281,1
30 PRINT"[CLR]"
35 TI$="000000"
40 PRINTTAB(14)"[RED][RVSON]MASTER[SPC]
MIND[RVSOFF]"
60 PRINT"[10CRSRD]":PRINT"[BLU][SPC]
REALIZADO[SPC]POR[SPC]J.[SPC]SALAS,P
ULSA[SPC]UNA[SPC]TECLA"
70 GET A$:IF A$=""GOTO 70
80 PRINT"[CLR][CRSRD][2SPC]EL[SPC]OR
DENADOR[SPC]ESCOGE[SPC]UN[SPC]NUMERO
[SPC]AL[SPC]AZAR."
90 M=0:N=0:B=0:RESTORE
100 PRINT"[2CRSRD]DE[SPC]CUANTOS[SPC]
TERMINOS[SPC]DESEA[SPC]EL[SPC]NUMERO
(1-6)"
105 INPUT A
107 IF A<1 OR A>6 THEN 80
109 D$="[4SPC]":FOR I=1 TO A:D$=D$+"
[BLK][SPC]":NEXT:FOR I=1 TO 40-2*A:D
$=D$+"[SPC]":NEXT
110 PRINT"[2CRSRD]DESEA[SPC]QUE[SPC]
SE[SPC]PUEDAN[SPC]REPETIR[SPC]TERMIN
OS"
120 INPUT"(S/N)";B$
125 IF B$<>"S" AND B$<>"N" THEN 120
130 PRINT:PRINT"PULSE[SPC]CUAL
QUIER[SPC]TECLA"
135 GET C$:IF C$="" GOTO 135
137 IF B$="S" THEN GOSUB1000:GOTO140

138 GOSUB2000
140 GOSUB 800:PRINT"[CLR]":GOSUB5000

150 M=M+1:GOSUB 7000
160 FOR I=1 TO A
170 GET C$:IF C$="" THEN 170
180 IF C$<"1" OR C$>"9" THEN 170
190 C(I)=VAL(C$) :PRINTC$"[SPC]";
200 NEXT
205 GOSUB 6000
210 GOSUB 3000:IF N=A THEN 300
220 PRINTN"M"B"H":PRINT
225 IF M>8 THEN PRINT"[HOM]":GOSUB50
00:PRINT"[16CRSRD]";
230 GOTO150
300 PRINT"[CLR][2CRSRD][3CRSRR][RED]
";:FOR I=1 TO A:PRINT LEFT$(STR$(B(I
)),2);:NEXT
340 PRINT:PRINT:PRINT"HECHO[SPC]EN[SPC]
";M;"INTENTOS":GOTO10000
345 GOSUB 8000
360 GET H$:IF H$=""GOTO 360
```

```
365 IF H$="S" GOTO 70
370 END
510 INPUT C$(I)
520 NEXT I
530 RETURN
800 POKE 54296,9:FOR Z=1 TO 7:READ L
(Z):NEXT
810 FORX=1TO14:READ D(X):NEXT:FORJ=1
TO7:POKE 54275,L(J):FOR Z=1 TO D(2*J
-1):NEXT
820 POKE54275,9:FOR Z=1 TO D(2*J):NE
XT Z
830 NEXT J
840 DATA 195,195,195,199,195,195,201

850 DATA 80,60,60,60,60,60,60,100,80
,60,60,80,80,80
860 RETURN
1000 FOR I=1 TO A:B(I)=INT(RND(0)*9)
+1:NEXT I:RETURN
2000 FOR I=1 TO A
2010 B(I)=INT(RND(0)*9)+1
2020 FOR Z=0 TO I-1:IF B(I)=B(Z) THE
N 2010
2030 NEXT Z:NEXT I
2040 RETURN
3000 B=0:N=0:FOR I=1 TO A:IF C(I)=B(
I) THEN N=N+1
3005 NEXT
3010 IF N=A THEN RETURN
3020 FOR I=1 TO A
3030 FOR J=1 TO A:IF C(I)=B(J) THEN
B=B+1
3040 NEXTJ:NEXT I:B=B-N:RETURN
5000 PRINT"[HOM][BLK]M=MUERTO[6SPC]H
=HERIDO[12SPC][RED]"D$"[BLU]":RETURN

6000 FOR I=1 TO 2:POKE 54296,3:POKE
54273,215
6010 FOR Z=1 TO 150:NEXT Z
6020 POKE 54273,0
6030 FOR Z=1 TO 150:NEXT Z
6040 NEXT I
6050 RETURN
7000 M$=STR$(M):PRINT MID$(M$,2,2)":
"TAB(4); RETURN
8000 POKE 54296,3: FOR I=1 TO 3 :POK
E54276,235
8010 FOR Z=1 TO 100:NEXT Z
8020 POKE54274,0
8030 FOR Z=1 TO 100:NEXT Z
8040 NEXT I
8050 POKE54295,220:FOR Z=15 TO 0 STE
P -1
8060 POKE54296,Z:FOR M=1 TO 100 :NEX
T M
8070 NEXT Z
8080 POKE 54295,0:POKE54296,0
8090 RETURN
9000 FORI=1TOA:PRINTB(I),C(I):NEXT
10000 PRINT"HAS[SPC]TARDADO[2CRSRR]"
MID$(TI$,3,2)"[2CRSRR]MINUTOS"
11000 I$=(MID$(TI$,5,2))
12000 PRINT  I$ "[2CRSRR]SEGUNDOS"
30000 PRINT"[CRSRD][13CRSRR][RVSON]H
A[SPC]CONSEGUIDO[RVSOFF]"
63901 PRINT"[CRSRD]"    (((A+5046)+(
A*100))-((M*100)+(I)+(Z)))"[CRSRR][RVSON]
PUNTOS[RVSOFF]"
63902 RR=(((A+5046)+(A*100))-((M*100
)+(I)+(Z)))
63905 IF RR>SC THEN PRINT"[RVSON]NUE
VO[SPC]RECORD[RVSOFF]":PRINT"[13CRSRR]
[CRSRU]"RR:SC=RR
63945 PRINT"[RVSON]RECORD[RVSOFF]"SC

63999 PRINT:PRINT"QUIERES[SPC]JUGAR[SPC]
OTRA[SPC]VEZ?[SPC](S/N)[BLU]":GOTO36
0
```



METEDURAS DE PATA

En nuestra página 53 del nº 7, se repite "del C-64, pero es difí" cuando debería decir "con un solo drive".

Líneas que faltan en "Nervios" de la sección Videocasino de nuestro número anterior.

```
400 IFFR=ROGOTO630
410 IFCA(1)>5000RCA(2)>500GOTO730
```

Errores corregidos del programa "auto" de la sección magia (C-64) del nº 7.

```
10 X=49152
20 READY:IFY=-1THEN40
30 POKEX,Y:X=X+1:Z=Z+Y:GOTO20
40 IFZ<>12374THENPRINT"ERROR[SPC]EN[SPC]
LAS[SPC]SENTENCIAS[SPC]DATA":END:REM
 SI Z ES DIFERENTE...
50 SYS49152:NEW
100 DATA169,90,133,251,169,0,133,252
,169,19,141,2,3,169,192,141,3,3,96,3
2,25
110 DATA192,76,134,164,24,169,10,101
,251,133,251,144,2,230,252,165,251,1
33,99
120 DATA165,252,133,98,162,144,56,32
,73,188,32,221,189,162,0,189,1,1,240
,9,32
130 DATA210,255,157,0,2,232,208,242,
32,18,225,201,13,240,3,76,105,165,56
,165
140 DATA251,233,20,176,2,198,252,169
,131,141,2,3,169,164,141,3,3,76,118,
165,-1
```

Errores de "Aventuras en el desierto" de nuestro nº 4 (VIC-20).

```
READY.

122 POKE38622,WC:POKE38624,WC

READY.

500 POKE36869,255:POKE36867,25:POKE36879,31

READY.
```



de las siguientes instrucciones:
10. POKES para protección de juego de caracteres.
20-60. Traslado de los punteros al juego de caracteres para posterior utilización de ellos.
70. Pixel de orientación del comecocos hacia la derecha.
80. Pixel de orientación del comecocos con la boca cerrada.
90. Pixel de orientación del comecocos hacia la izquierda.
100. Pixel de orientación del comecocos hacia abajo.
110. Pixel de orientación del comecocos hacia arriba.
120. Pixel de creación de muros del laberinto.
130. Pixel de creación de bombas.
140. Pixel de creación de monstruos.
150. Pixel de creación de espacios en blanco.
160. Pixel de creación de bombas.
170. Pixel de creación de puntitos de puntos.
200. Data final.
210-280. Instrucciones.
290. Carga automática de la 2ª parte y vuelta a la normalidad de caracteres.

2) Es la parte donde se desarrolla todo el juego:

430. Indica los bytes libres de que se dispone.
450-470. Variables principales. 460=variables para utilización del joystick.
490. Subrutina de presentación.
500. Conversión al juego de caracteres y función de teclas repetitivas.
510. Color de fondo-pantalla: blanco.
550-640. Creación del laberinto y de los puntitos.
700. Pregunta al teclado.
660. Bombas iniciales. Van aumentando.
720. Si consigues 2.200 puntos y te comes aquel signo, pasas de pantalla. Luego debes conseguir otros 2.200 puntos y así sucesivàmente.

```
1 REM* PRIMERA PARTE *
5 REM*LABYRINTH PACK*
10 POKE52,28:POKE56,28
20 FORT=7168TO7249:POKET,PEEK(T+2560
0):NEXT
30 POKE36869,255
40 READA:IFA=-1THEN200
50 FORT=0TO7:READB,POKE7168+A*8+T,B:
NEXT
60 GOTO40
70 DATA50 ,60,126,118,126,112,112,12
6,60
80 DATA51 ,60,126,118,126,126,126,12
6,60
90 DATA52 ,60,126,118,126,14,14,126,
60
100 DATA53 ,126,255,255,255,247,207,
207,78
110 DATA54 ,78,207,207,247,255,255,2
55,126
120 DATA55 ,255,255,255,255,255,255,
255,255
130 DATA56 ,68,40,16,56,124,124,124,
56
140 DATA57 ,28,62,127,93,127,127,127
,85
150 DATA58 ,0,0,0,0,0.0,0,0
160 DATA59 ,16,16,56,56,56,56,124,12
4
170 DATA60 ,0,0,0,16,0,0,0,0
200 DATA-1
210 PRINT"[CLR]"
220 PRINT"[RVSON][RED]****LABYRINTH[SPC]
PACK****[BLU]"
230 PRINT"[CRSRD][4CRSRR][RVSON][BLK]
=[SPC]DERECHA[RVSOFF]::";2
240 PRINT"[CRSRD][4CRSRR][RVSON]:[SPC]
IZQUIERDA[RVSOFF]";4
250 PRINT"[CRSRD][4CRSRR][RVSON]SPC[2SPC]
ABAJO[RVSOFF]:";5
```

730. Variables para la utilización de joystick.
740. Control de packman extra.
800-850. Controla la posición de packman y sus choques contra los muros.
900-970. Control de contelleo de packman y sus movimientos.
980-995. Indica los puntos y cronometra el tiempo de aparición de los monstruos. Cada vez aparecen un seg. menos. 995=control de retardo de aparición.
2000. Subrutina de sonido de choque contra un muro.
2500. Subrutina de sonido cuando come un puntito y contador de puntos.
3000. Subrutina de aparición de monstruos, hasta la línea 4.060, junto con la aparición de bonus extra —4000, 4020— y de bombas —4060—. Si la salida de un monstruo coincide con la posición de una bomba o de packman pasa a la subrutina 4500.
4500-4580. Subrutina de sonido de destrucción de un packman. Si quedan 0 packman va a la línea 6000.
4995. Primera subrutina de cambio de pantalla.
5000-5120. Subrutina de presentación y de cambio de pantalla. CREDITS.
6000-6090. Subrutina de fin de juego.
6500. Subrutina de sonido de packman extra, —son cada 7500 P—. Abarca hasta la línea 6540.
7000-7040. Subrutina de sonido cuando comes un bonus extra.

Como pueden apreciar, este programa está realizado con gráficos definidos por mí, por lo que quiero felicitar sinceramente al equipo de COMMODORE WORLD por haber puesto un artículo sobre la falta de resolución en el VIC-20. Sin embargo quiero aportar "algo", si se puede llamar así, aportando un tema a tratar en la revista, y valga la redundancia; la utilización del cartucho del SUPER EXPANDER en el VIC.

```
260 PRINT"[CRSRD][4CRSRR][RVSON]@[SPC]
ARRIBA[RVSOFF]:::";6
270 PRINT"[3CRSRD][3CRSRR][RVSON]PUL
SAR[SPC]UNA[SPC]TECLA[RVSOFF]"
280 GETA$:IFA$=""THEN280
290 PRINT"[CLR][CRSRD][SPC]CARGANDO[SPC]
ALTA[SPC]RESOLU[2SPC][CRSRD]CION[CRSRD]
":POKE631,131:POKE198,1:POKE36869,24
0
```

SEGUNDA PARTE

```
430 PRINTFRE(0);"BYTES[SPC]FREE":FOR
T=1TO1000:NEXT:REC=0
450 SC=0:HU=0:LI=3:TY=1:LK=7500:CR=1

460 POKE37151,0:PA=37151:PB=37152:RB
=37154
470 X=1:Y=19:SS=30720:TI$="000000":Q
W=7680:MO=50:VO=36878:S3=36876
490 GOSUB5000
500 POKE36869,255:POKE650,128
510 POKEVO+1,25
550 PRINT"[CLR][HOM][CRSRD][BLK]7777
777777777777777777";
570 PRINT"7<<<<<<<<<7<<<<<77<77<77
77<7<7<7<77<7<<<<77<<<<7<7<<<<<<<<7<
77<<";
580 PRINT"<<77<7<77<7<777<<<7<<77<<<
<<<<<7<<<<<7777<77<777<777<7<777<<<
<<<7";
590 PRINT"7<<<7<<<<<7<<<<777<7<77<<<
77<7777777<<<<<7<777<<<<<<<<7<<<<777
77<7";
600 PRINT"7<<7<<<7<<7<77<7<<<<<77<77
77<77<7<7<<7<777<77<77<<<<<<7<77<7<<
<7<7";
610 PRINT"7<<7<7777<<<<<<<<7<7<77<7<
<<<7<<77777<<<<<<77<<<7<77<<<<<<7<7<
<<77";
620 PRINT"7<<<7<<<<7777<7<7<7<<<<<7<
7777<7<<<<<<777<777<7<7<<<<77<777<<<
<<<7";
630 PRINT"7<<<<<77<<<<<<<<<777<<7";
640 PRINT"7777777777777777777777";
660 FORF=1TOTY:FG=INT(RND(1)*416)+44
:POKEQW+FG,59:POKESS+QW+FG,2:NEXT
700 GETA$
710 POKE7725,0
720 IFSC>=2200+IOANDPEEK(QW+X+Y*22)=
0THENGOSUB4995:GOTO460
730 A=PEEK(PA):POKERB,127:B=PEEK(PB)
:POKERB,255
740 IFSC>=LKTHENGOSUB6500
800 IF(AAND4)=0THENY=Y-1:MO=54
805 IFPEEK(QW+X+Y*22)=55ORPEEK(QW+X+
Y*22)=59THENY=Y+1:GOSUB2000
810 IF(AAND8)=0THENY=Y+1:MO=53
815 IFPEEK(QW+X+Y*22)=55ORPEEK(QW+X+
Y*22)=59THENY=Y-1:GOSUB2000
820 IF(AAND16)=0THENX=X-1:MO=52
825 IFPEEK(QW+X+Y*22)=55ORPEEK(QW+X+
Y*22)=59THENX=X+1:GOSUB2000
830 IF(BAND128)=0THENX=X+1:MO=50
835 IFPEEK(QW+X+Y*22)=55ORPEEK(QW+X+
Y*22)=59THENX=X-1:GOSUB2000
850 IFPEEK(QW+X+Y*22)=60THENGOSUB250
0
860 IFPEEK(QW+X+Y*22)=56THENGOSUB700
0
910 POKEQW+X+Y*22,MO
920 POKESS+QW+X+Y*22,6
950 FORT=1TO100:NEXT
960 POKEQW+X+Y*22,51
970 FORT=1TO100:NEXT
980 PRINT"[HOM][RVSON][GRN]SCORE:";S
C;"[SPC]TIME:";VAL(TI$)
990 IFVAL(TI$)=30-HTHENGOSUB3000:H=H
+1
995 IFH>=15THENH=0
1000 POKEQW+X+Y*22,58
1500 GOTO700
2000 POKEVO,15:POKES3,240:POKES3,0:R
ETURN
2500 POKEVO,15:POKES3-1,255:POKES3-1
,0:SC=SC+10:RETURN
3000 FORMN=1TO5+AU
3010 AA=INT(RND(1)*416)+44
3020 IFPEEK(QW+AA)=55THEN3090
3030 IFPEEK(QW+AA)=59ORPEEK(QW+AA)=5
1THEN4500
3040 POKEQW+AA,57:POKEQW+SS+AA,5
3060 POKEVO,15:FORNN=150TO250:POKES3
,NN:NEXT:POKES3,0
3070 FORT=1TO500:NEXT
3080 POKEQW+AA,58
3090 NEXT
4000 AB=INT(RND(1)*10)
4010 HU=INT(RND(1)*416)+44
4020 IFAB=5ORAB=6THENPOKEQW+HU,56:PO
KEQW+SS+HU,4
4050 AU=AU+1:PRINT"[HOM][RVSON][GRN]
SCORE:";SC,"[SPC]TIME:[4SPC]":TI$="0
00000"
4055 IFAU>=15THENAU=0
4060 POKEQW+AA,59:POKESS+QW+AA,4:RET
URN
4500 POKES3+1,220
4505 POKEQW+AA,57
4510 FORRE=15TO0STEP-1
4520 POKEVO,RE
4530 FORT=1TO300:NEXT
4540 NEXTRE
4550 POKES3+1,0
4560 LI=LI-1:IFLI=0THEN6000
4570 POKEQW+X+Y*22,58:X=1:Y=19:TI$="
000000"
4575 POKEQW+AA,58
4580 RETURN
4995 CR=CR+1:TY=TY+2:IO=IO+2200:RETU
RN
5000 POKE36869,240:PRINT"[CLR]":POKE
VO+1,110
5010 FORRE=1TO10:FORT=1TO250:NEXT
5020 PRINT"[CLR]":FORT=1TO210:NEXT:P
RINT"[WHT]C.ROJAS[SPC]PRESENT:[HOM][10CRSRD]
[6CRSRR]SCREEN:";CR:NEXT
5025 PRINT"[CLR][10CRSRD][RVSON][YEL]
****LABYRINTH[SPC]PACK****[BLU][RVSOFF]
":FORT=1TO5000:NEXT
5120 TI$="000000":RETURN
6000 POKE36869,240:POKE36879,105
6010 PRINT"[CLR][2CRSRU][RVSON][YEL]
****LABYRINTH[SPC]PACK****[BLU][RVSOFF]
"

6020 PRINT"[CRSRD][WHT]SCORE:";SC
6030 IFSC>RECTHENREC=SC
6040 PRINT"[CRSRD][WHT]RECORD:";REC
6050 PRINT"[CRSRD][WHT]CREDIT:";CR
6060 PRINT"[CRSRD][WHT]ANOTHER[SPC]G
AME[SPC](S/N)[SPC]?"
6070 GETA$:IFA$=""THEN6070
6080 IFA$="S"THEN450
6090 END
6500 FORJJ=1TO25:POKEVO,15:POKES3,24
0
6510 FORT=1TO25:NEXT
6520 POKES3,0
6530 FORT=1TO25:NEXT:NEXT
6540 LK=LK+7500:LI=LI+1:RETURN
7000 POKEVO,15
7010 FORLL=1TO100:POKES3,INT(RND(1)*
128)+128
7020 FORT=1TO10:NEXT:NEXT
7030 POKEVO,0:POKES3,0
7040 SC=SC+500:RETURN
```



C-64

Astral

Ignacio Ortega Ortigosa
Virgen del Carmen, 19 d/2º B
Algeciras - Cádiz

Estas líneas van dirigidas más especialmente a los que, como yo, sois novatos en el arte de ordenar ordenadores. Los otros, los de mayor y mejor nivel, también pueden leerlas, pero que sean indulgentes con lo que para ellos pueden ser herejías informáticas.

A la vista de las cifras de venta de máquinas CBM, pienso que no soy el único novel. A mí me pusieron los Reyes un 64 y entré en faena. También me suscribí al encantador Mundo del Commodore y empezaron las sorpresas.

Algunas personas me decían que un ordenador es una máquina tonta a la que hay que enseñarle todo. Primera sorpresa: La máquina no es tonta; la máquina es inflexible. Y me llamaba tonto a mí, con la delicada frase: Syntax error... ¿A vosotros también?

Pero no hay que desanimarse. Adelante. Comprendiendo o más bien intuyendo que el Manual del Usuario es un librito escueto con el que se consigue poca cosa más que cambiar grados centígrados a fahrenheit y unas insulsas pelotas rebotando, pasamos a adquirir la Guía de Referencia, ¡oh maravilla! 3.000 pelas. Ahí empezamos a entrever que hay otro mundo detrás de otras puertas.

Los autodidactas continuamos insistiendo. E intentado aprender. En éstas, el cartero llama y nos trae el primer número de Commodore World. Como fieras transcribimos a nuestro teclado los progamas. ¡Caramba!, ¿pero se pueden hacer cosas como el Editor de Sprites de José Ramón Lasa? ¿Cuándo llegaré yo a eso? Leemos el Editorial, en profundidad y nos llega al alma sobre todo el párrafo de crear un Club de Amigos de todas las edades y de admitir cualquier idea o programa más simple.

¡Ah! Esto es muy bueno. ¿Debería yo enviar ese programilla que soluciona los problemas de velocidad, tiempo y espacio?... A lo mejor ya está muy visto... Reflexionamos que es posible que nuestro mejor logro, nuestro mayor éxito en programación de novatos está ya más que hecho y superado. Y continuamos el estudio de la Guía de Referencia, con la pregunta en pie. ¿Envío o no envío el programilla?

La Guía, entretanto, se encarga de amargarnos. Ya está hablando de cosas como Selección de Banco y Adaptador complejo de interface. ¡Qué barbaridad! Bueno, nosotros a lo nuestro. Vamos a copiar el ejemplo de los dibujos en alta resolución a ver qué pasa. ¡Sopla!... Ya me he cargado el ordenador. Pero no. Aquí dice que la pantalla se llena de porquería. Entonces debe ser normal. Menos mal que el libro ha avisado. Sigamos. Ahora la curva del seno. Pues no funciona. Tardamos algún tiempo en comprender que los puntos de la sinusoide deben esperar su turno para salir. El libro debería haberlo advertido para no darnos otro susto. Ahora entendemos por qué nos decían que el Basic es lento, cuando observábamos que era capaz de imprimir la tabla de multiplicar en un periquete.

En éstas llega el número 2 de la Revista. ¡Qué voy a enviar yo mi miserable programilla! Menudo listado tiene Manuel Torralba a sus trece años con el Topo Loco manejando pokes y peeks a todo plan. Y, entretanto, la Guía de Referencia haciendo de las suyas. En la página 88, después de hacernos teclear ocho sentencias, nos dice textualmente: ¡Usted acaba de crear un signo @ en color invertido!... Pues no veo yo...

En fin, otra vez en alta resolución y después de averiguar que al libro le falta un next, pintamos un semicírculo en la pantalla... por fin.

No nos desanimemos amigos, sigamos insistiendo en el estudio y machacando teclado, que siempre algún fruto sacaremos.

Y como ayuda-guía os envío este programilla que bueno o malo, funciona, dedicado especialmente a aquellos que sin grandes conocimientos ni experiencia como yo, se han desilusionado en principio al ver que el ordenador no perdona cuando te equivocas.

Me anima enviarlo también el ver que poco se ha tocado hasta ahora en las posibilidades gráficas del 64 y que con los dos ejemplos del libro tampoco hay para mucho.

Siento no poder prestar un arreglito para los Vic-ciosos del 20, pero mis modestos conocimientos me incapacitan para eso.

Novatos... animaros a enviar cosas. Aunque os parezcan superadas por hace ya tiempo, nos pueden servir a los demás principiantes.

Y ahora al trabajo:

Rutina 1.—Líneas 1000 a 1040. Todo esto está calcado de la Guía de Referencia. Es lo que nos asustó cuando lo pusimos en marcha la primera vez. Coge la pantalla y la hace un churro pero eso sí con un divertido coloreado. Luego se formaliza y es como si quisiera dos telones. Podéis cambiar en la línea 1030 en el pokel que está en 3, el código de vuestro color favorito (negro no vale).

Rutina 2.—Líneas 1100 a 1115. También fusiladas de la Guía. Es la fórmula que da el libro para pintar un punto determinado en la pantalla y encima diciéndote que el Basic puede invadirlas.

Rutina 3.—Líneas 2000 a 2350. Es el lío que me he inventado para pintar líneas a voluntad punto a punto. Seguramente se podrá hacer con menos instrucciones o por otro procedimiento más fácil. No lo sé y me da igual, porque éste funciona y es mejor ▶

que el del libro que deja puntos separados en algunas zonas. Este pinta rectas y, con un poco de astucia, unas curvas preciosas.

Programa para utilizar estas rutinas:

Línea 20.—Sitúa el centro del dibujo en el centro de la pantalla. Si mueves esos valores el dibujo sale en el punto elegido.

Líneas 70 y 80.—Se hace el reparto de vértices simétricos y determinan la dirección del segmento.

Y todas las demás hacen el cálculo de los puntos y direcciones que han de llevar de una forma continua.

Podéis volver a la pantalla normal con Stop y Run/stop-restore.

Si queréis el ambiente más cargado podéis añadir:

225 NP = NP + 1 : GOTO80

La coordenada vertical tiene 200 puntos, por lo que os aconsejo que el radio exterior no supere el número 100 porque si no el dibujo queda incompleto. Pero si lo hacéis, tampoco se rompe nada (veréis que esto de las averías me preocupa mucho y es que los Reyes están todavía pagando las letras).

Me gustaría que Pere Masats, artista neurocirujano informático, nos fabricase unas rutinas en CM para que todo este lío de cosas fuese menos lento, especialmente las rutinas 1 y 2 que son los desesperantes cambios de decorado.

Tengo otras cositas monas sobre el mismo tema. ¿Las queréis? No dejéis de trabajar, insisto, y mandad lo que tengáis por tonto que sea. Yo he aprendido más de las cosas de otros, sobre todo las facilonas, que del libro.

```
1 REM..........ASTRAL...............
....
6 :
7 :
8 :
10 PRINT"[CLR]"
20 CX=160:CY=100
30 INPUT"[3CRSRD][2CRSRR]NUMERO[SPC]
DE[SPC]PICOS";NP
40 INPUT"[2CRSRD][2CRSRR]RADIO[SPC]E
XTERIOR";RM
50 INPUT"[2CRSRD][2CRSRR]RADIO[SPC]I
NTERIOR";RN
60 GOSUB1000
70 AN=2*π/NP
80 DA=π/NP
90 X1=CX+RM*COS(AN):Y1=CY+RM*SIN(AN)

100 FORI=1TONP
110 AN=AN+DA
120 X=CX+RN*COS(AN):Y=CY+RN*SIN(AN)
130 DX=X-X1:DY=Y-Y1
140 X=X1:Y=Y1
150 GOSUB2000
160 X1=X:Y1=Y:AN=AN+DA
170 X=CX+RM*COS(AN):Y=CY+RM*SIN(AN)
180 DX=X-X1:DY=Y-Y1
190 X=X1:Y=Y1
200 GOSUB2000
210 X1=X:Y1=Y
220 NEXTI
230 GOTO230
999 END
1000 REM      PASE A  BIT MAPPED
1001 REM    "[20COMMY]"
1005 BASE=2*4096:POKE53272,PEEK(5327
2)OR8
1010 POKE53265,PEEK(53265)OR32
1020 FORI=BASETOBASE+7999:POKEI,0:NE
XT
1030 FORI=1024TO2023:POKEI,3:NEXT
1040 RETURN
1042 :
1098 REM.....CALCULO DEL PUNTO.....
1099 REM    "[21COMMY]"
1100 CH=INT(X/8)
1110 RO=INT(Y/8)
1120 LN=YAND7
1130 BY=BASE+RO*320+8*CH+LN
1140 BI=7-(XAND7)
1150 POKEBY,PEEK(BY)OR(2↑BI)
1155 RETURN
1998 REM      CALCULO DEL SEGMENTO
1999 REM"[2SPC][23COMMY]"
2000 X1=X:Y1=Y
2010 X2=X1+DX:Y2=Y1+DY
2015 IFABS(DX)>ABS(DY)THEN2200
2020 IFDY<0THEN2100
2030 FORS=Y1TOY2
2040 GOTO2110
2080 RETURN
2100 FORS=Y1TOY2STEP-1
2110 X=(X1+DX/(-DY)*(Y1-S))
2120 Y=S
2130 GOSUB1100
2140 NEXTS
2150 RETURN
2200 IFDX<0THEN2300
2230 FORS=X1TOX2
2240 GOTO2310
2280 RETURN
2300 FORS=X1TOX2STEP-1
2310 Y=(Y1+DY/(-DX)*(X1-S))
2320 X=S
2330 GOSUB1100
2340 NEXTS
2350 RETURN
```



Asterisco

VIC 20
SIN AMPLIACION

Higinio Porta López - 16 años
Avenida Huesca, 20
Sariñena (Huesca)

Como es el primer programa un poco largo que hago, tal vez lo encontréis un poco liado y desordenado; pero como funciona, ya vale.

El programa ocupa casi toda la memoria del VIC, deja 36 bites libres. Es una pena que el VIC tenga tan poca memoria. Por esa falta de memoria no he podido hacer la presentación mejor.

Ahora paso a explicaros de qué va:

Es un juego parecido a los de marcianitos, pero no os asustéis, cambia en algunas cosas. (Espero que no hayáis aborrecido ya a los pobres marcianitos.)

Manejas la nave por toda la pantalla persiguiendo al marcianito —sólo hay uno—. Debes disparar y darle, es decir, hacer blanco en él.

Al principio dispones de 3 tankes (naves) y 8 tiros —que puedes aumentar— por cada tanke. Cada vez que le das consigues 10 puntos y un tiro más; si fallas, lógicamente, pierdes el tiro. Cuando agotas los tiros pierdes el tanke, y si agotas los tankes...

Todo esto se explica en la presentación del programa, pero aún hay más: A los 200, 400 y 800 puntos consigues 2 tankes extras. ¡Ah!, y si consigues el récord, pones tu nombre.

Nada más, sólo me queda animar a todos para que manden colaboraciones. Yo creo que ayudan mucho a los principiantes.

```
1 A4$="[SPC]NADIE"
2 POKE36879,25
5 POKE36878,15
6 GOSUB11000
10 G=7800:L=8130:A3=0
20 TN=3:TR=8
30 AR=113:AB=114:IZ=115:DE=107
40 PT=0
45 PRINT"[CLR]"
50 FORP=7724TO7745:GOSUB100:NEXT
60 FORP=7746TO8164STEP22:GOSUB100:NE
XT
70 FORP=8165TO8185:GOSUB100:NEXT
80 FORP=8163TO7767STEP-22:GOSUB100:N
EXT
90 GOTO110
100 POKEP,102:POKEP+30720,4:RETURN
110 PRINT"[HOM]QUEDAN";TN;"TANKS";TR
;"[SPC]TIR"
120 PRINT"[HOM][CRSRD]PTS";PT;"[3SPC]
REC[SPC]PTS";HP
130 POKEG,42:POKEG+30720,0
140 POKEL,AR:POKEL+30720,0
1000 POKE650,255
1010 GETA$:IFA$=""THEN2000
1020 IFA$="L"THENC=1:POKEL,32:L=L-22
:GOSUB7000
1030 IFA$=","THENC=2:POKEL,32:L=L+22
:GOSUB7000
1040 IFA$="Z"THENC=3:POKEL,32:L=L-1:
GOSUB7000
1050 IFA$="X"THENC=4:POKEL,32:L=L+1:
GOSUB7000
1060 IFA$="[SPC]"THENGOSUB7010:GOSUB
3000
1070 IFC=1THENIFPEEK(L)=102THENL=L+2
2:GOSUB7020
1080 IFC=2THENIFPEEK(L)=102THENL=L-2
2:GOSUB7020
1090 IFC=3THENIFPEEK(L)=102THENL=L+1
:GOSUB7020
1100 IFC=4THENIFPEEK(L)=102THENL=L-1
:GOSUB7020
1110 IFPEEK(L)=42THENPOKEL,42:GOSUB7
040:GOSUB4200:GOSUB4010:L=8130:POKEL
,AR
1120 IFC=1THENPOKEL,AR:POKEL+30720,0

1130 IFC=2THENPOKEL,AB:POKEL+30720,0

1140 IFC=3THENPOKEL,IZ:POKEL+30720,0

1150 IFC=4THENPOKEL,DE:POKEL+30720,0

2000 POKEG,32
2010 F=INT(RND(1)*5+1)
2020 IFF=1THENG=G-22:IFPEEK(G)=102TH
ENG=G+22
2040 IFF=2THENG=G+22:IFPEEK(G)=102TH
ENG=G-22
2060 IFF=3THENG=G-1:IFPEEK(G)=102THE
NG=G+1
2080 IFF=4THENG=G+1:IFPEEK(G)=102THE
NG=G-1
2100 B=PEEK(G)
2110 IFB=107ORB=113ORB=114ORB=115THE
NPOKEG,42:GOSUB7040:GOSUB4200:L=8130
:GOSUB4010:POKEL,AR
2120 POKEG,42:POKEG+30720,0:POKE3687
4,246:FORU=1TO20:NEXT:POKE36874,0:GO
TO1000
3000 N=L:Y=0
3010 K=PEEK(L)
3020 IFK=ARTHENM=-22
3030 IFK=ABTHENM=22
3040 IFK=IZTHENM=-1
3050 IFK=DETHENM=1
3060 S=PEEK(N+M)
3070 IFS=102THENGOSUB7050:POKEN,32:G
OSUB4000:RETURN
3080 IFS=42THENPOKEN,32:GOSUB6000:TR
=TR+1:GOSUB7030:GOSUB4010:RETURN
3090 Y=Y+1
3100 IFY<>1THENPOKEN,32
3110 IFK=ARORK=ABTHENPOKEN+M,93:POKE
N+M+30720,0
3120 IFK=IZORK=DETHENPOKEN+M,64:POKE
N+M+30720,0
3130 N=N+M
3140 GOTO3060
4000 TR=TR-1:IFTR=0THENTN=TN-1:GOSUB
7040:TR=8:POKEL,32:L=8130:POKEL,AR:G
OSUB4100
4010 PRINT"[HOM][14CRSRR]";TR;"[CRSRL]
[SPC]":RETURN
4100 IFTN=0THENPOKE36874,0:GOTO5000
4110 PRINT"[HOM][6CRSRR]";TN;"[CRSRL]
[SPC]":PRINT"[HOM][9CRSRR]T":RETURN
4200 TN=TN-1:TR=8:IFTN=0THENPOKE3687
5,0:GOTO5000
4210 PRINT"[HOM][6CRSRR]";TN;"[CRSRL]
[SPC]":PRINT"[HOM][9CRSRR]T":RETURN
5000 PRINT"[CRSRU][7CRSRR]0[7CRSRR]0
[SPC]"
5001 PRINT"[9CRSRD][6CRSRR]GAME[SPC]
OVER":FORU=1TO4000:NEXT:GOSUB10000
5005 PRINT"[CLR][CRSRD][CRSRR]JUEGAS
[SPC]MAS?":PRINT"[HOM][3CRSRD][CRSRR]
(S/N)":POKE36878,0
5010 GETZ$:IFZ$=""THEN5010
5020 IFZ$="S"THEN2
5030 IFZ$="N"THENEND
5040 GOTO5010
5999 REMPUNTOS********
6000 PT=PT+10:IFPT>HPTHENGOSUB9000:H
P=PT:GOSUB6100
6005 IFPT=200THENGOSUB8000:POKE36879
,24:TN=TN+2:GOSUB4210
6007 IFPT=400THENGOSUB8000:POKE36879
,31:TN=TN+2:GOSUB4210
6008 IFPT=800THENGOSUB8000:POKE36879
,88:TN=TN+2:GOSUB4210
6010 PRINT"[HOM][CRSRD][3CRSRR]";PT:
RETURN
6100 PRINT"[HOM][CRSRD][17CRSRR]";HP
:RETURN
7000 POKE36875,230:FORU=1TO10:NEXT:P
OKE36875,0:RETURN
7010 POKE36877,237:FORU=1TO50:NEXT:P
OKE36877,0:RETURN
7020 POKE36876,150:FORU=1TO20:NEXT:P
OKE36876,0:RETURN
7030 FORU=15TO0STEP-1:POKE36876,232+
U:FORR=1TO10:NEXTR,U:POKE36876,0:RET
URN
7040 FORO=-30TO55:POKE36876,200+O:NE
XTO:POKE36876,0:RETURN
7050 POKE36874,242:FORU=1TO50:NEXT:R
ETURN
8000 PRINT"[HOM][13CRSRD][8CRSRR]BON
US"
8010 FORA1=1TO30:POKE36876,INT(RND(1
)*128)+128:FORA2=1TO10:NEXTA2,A1:POK
E36876,0
8020 PRINT"[HOM][13CRSRD][8CRSRR][5SPC]
":RETURN
9000 A3=A3+1:IFA3=1THENIFHP<>0THENPR
INT"[HOM][13CRSRD][5CRSRR]NUEVO[SPC]
RECORD":GOTO9010
9005 RETURN
9010 FORA1=1TO50:POKE36876,INT(RND(1
)*128)+128:FORA2=1TO10:NEXTA2,A1:POK
E36876,0
9020 :PRINT"[HOM][13CRSRD][5CRSRR][12SPC]
":RETURN
10000 IFRP<HPTHENRP=HP:GOTO10020
10010 IFRP<HPTHENRP=HP:GOTO10020
10015 GOTO10100
```

```
10020 POKE36878,0:PRINT"[CLR][CRSRD]
[CRSRR]TU[SPC]TIENES[SPC]EL[SPC]RECO
RD!":POKE36879,156
10030 PRINT"[HOM][3CRSRD][CRSRR]ESCR
IBE[SPC]TU[SPC]NOMBRE"
10040 INPUT"[HOM][5CRSRD][CRSRR]";A4
$
10050 PRINT"[CLR][CRSRD][6CRSRR]RECO
RDMAN:"
10060 PRINT"[HOM][9CRSRD][7CRSRR]";A
4$
10070 FORU=1TO5000:NEXT
10080 RETURN
10100 PRINT"[CLR][CRSRD][CRSRR]LO[SPC]
SIENTO,[SPC]NO[SPC]TIENES":PRINT:PRI
NT"[8SPC]RECORD":POKE36879,154
10110 PRINT:PRINT"[HOM][5CRSRD][7CRSRR]
LO[SPC]TIENE:[CRSRD]"
10120 PRINT"[HOM][7CRSRD][7CRSRR]";A
4$:FORU=1TO5000:NEXT
10130 RETURN
11000 PRINT"[CLR][CRSRD][3CRSRR]HAZ[SPC]
BLANCO[SPC]EN[SPC]EL"
```

```
11010 PRINT"[CRSRD][6CRSRR]ASTERISCO
"
11020 PRINT"[CRSRD][CRSRR]Z=IZDA.";
11030 PRINT"[5SPC]X=DECHA."
11040 PRINT"[CRSRD][CRSRR]L=ARRIBA[4SPC]
,=ABAJO"
11050 PRINT"[CRSRD][3CRSRR]ESPACIO=D
ISPARO"
11060 PRINT"[CRSRD][CRSRR]TIENES[SPC]
3[SPC]TANKES[2SPC]Y"
11070 PRINT "[CRSRD][CRSRR]8[SPC]TIR
OS[SPC]POR[SPC]TANKE"
11080 PRINT"[CRSRD][CRSRR]CADA[SPC]B
LANCO:"
11090 PRINT"[CRSRD][7CRSRR]1[SPC]TIR
O[SPC]MAS"
11100 PRINT"[CRSRD][6CRSRR]10[SPC]PU
NTOS[SPC]MAS
11110 PRINT"[CRSRD][3CRSRR]PULSA[SPC]
UNA[SPC]TECLA"
12000 GETA6$:IFA6$=""THEN12000
12001 RETURN
```



Prints sofisticados

C-64

José Luis Muñoz Franco
Monterde, 1-7º B
Zaragoza - Tel. 34 00 70

Os remito un programa, en realidad tres subrutinas, que creo que servirán como complemento al artículo publicado en el número 3 de abril titulado PRINTS SOFISTICADOS. Están hechas para su uso en un Commodore 64, aunque creo que son fácilmente adaptables al Vic. Lo primero que quiero puntualizar es que no he querido agrupar líneas de programa mediante la técnica de los dos puntos para hacer más fácil su comprensión, por lo que cada uno es libre de agrupar las líneas que pueda. Así mismo he separado lo que son palabras Basic y variables dejando un espacio entre ambas, también por el motivo anterior. Y paso ya a explicar un poco el programa.

Comienza en la línea 10 solicitando Print 1, 2 ó 3.

Línea 20: Filtra los caracteres que no sean 1, 2 ó 3 no dejándolos pasar.
Línea 25: Pide dar entrada a una cadena.
Línea 30: Desvía la ejecución del programa hacia una de las subrutinas elegidas.
Línea 40: Bucle de tiempo antes de volver a ejecutar de nuevo el programa.

El programa no tiene más. Ahora paso a explicar las subrutinas.

NUMERO 1: IMPRIME ALTERNANDO UN CARACTER DE IZQUIERDA Y OTRO DE LA DERECHA DE LA CADENA.

Línea 1010: Mide la longitud de la cadena.
Línea 1020: Si la longitud es impar añade un espacio por la izquierda a la cadena. Si es par añade un espacio por la izquierda y otro por la derecha.
Línea 1030: Inicia un bucle para imprimir los caracteres de la cadena.
Línea 1040 a 1070: Se encargan de imprimir un carácter de la izquierda y desplazan el cursor los lugares necesarios a la derecha.
Línea 1080 a 1090: Imprimen carácter de la derecha.
Línea 1095: Vuelve para imprimir otros dos caracteres.

Si os habéis fijado, la subrutina no trabaja imprimiendo los caracteres de uno en uno sino que primero imprime uno de la izquierda y uno de la derecha, luego 2 de la izquierda y dos de la derecha, etc...

NUMERO 2: DESPLAZA CADA CARACTER DE LA CADENA A LO LARGO DE UNA LINEA Y LUEGO REPITE POR LA LINEA SIGUIENTE HASTA QUE EL CARACTER OCUPA SU LUGAR

Línea 2010: Inicia el bucle para la impresión de todos los caracteres.
Línea 2020: Si el carácter es un espacio establece V en cero para acelerar el bucle de tiempo de la línea 3060 y 2060.
Línea 2040 a 2080: Desplazan el carácter correspondiente por la línea superior.
Línea 3000 a 3080: Desplazan el carácter por la línea inferior después de haber bajado una línea el cursor en 2090.
Línea 3090: Imprimen el carácter definitivamente.
Línea 4000: Sitúa el cursor para en la línea 4010 volver a tomar otro carácter.

Creo que se puede adaptar para el VIC si en la línea 2040 ponemos 21 y no 39. En la 3000 ponemos lo mismo.

NUMERO 3: IMPRIME UNA CADENA. LUEGO TOMA CADA CARACTER Y LO VA DESPLAZANDO POR LA LINEA DE ARRIBA HASTA QUE LO COLOCA DELANTE DE LA CADENA QUE HA IMPRIMIDO.

Línea 5010: Mide la cadena y asigna a C un valor que es el del doble de la longitud de cadena.
Línea 5020: Imprime la cadena en el lugar correspondiente.
Línea 5030: Establece bucle para impresión de caracteres.
Línea 5040: Sitúa cursor sobre carácter a desplazar, lo borra y lo sitúa en la línea superior.
Línea 5050: Establece bucle para desplazar el carácter.
Línea 5060: Imprime carácter.
Línea 5070: Bucle de tiempo.
Línea 5080: Borra arácter del lugar en que está, para en la 5090, seguir con el bucle de desplazamiento.
Línea 6000: Sitúa definitivamente el carácter en el lugar que le corresponde. A la variable C que se utilizará en la línea 5040 para situar el cursor sobre el siguiente carácter a desplazar, le resta 1.
Línea 6010: Vuelve para tomar otro carácter.

Y nada más, me despido esperando que os hayan gustado estas tres subrutinas.

```
5 REM    ************************
6 REM    *** SUBRUTINAS PRINT ***
7 REM    *** POR J.LUIS MUNOZ ***
8 REM    ************************
10 PRINT"[CLR]":PRINT "[9SPC]SOLICIT
A[SPC]PRINT[SPC]1-2-3"
20 GET C$:IFC$= "1" OR C$= "2" OR C$
= "3" THEN25
22 GOTO 20
25 INPUT"[2CRSRD][9SPC]UNA[SPC]PALAB
RA[SPC]";A$
27 PRINT"[2CRSRD]"
30 ON VAL(C$)GOSUB 1000,2000,5000
40 FOR L=1 TO 600:NEXT L:GOTO 10
1000 REM SUBRUTINA PRINT NUMERO 1***
************************************
*******
1010 L=LEN(A$)
1020 IFL/2 <> INT(L/2) THEN A$="[SPC]
"+A$:L=L+1:GOTO 1030
1025 A$="[SPC]"+A$+"[SPC]":L=L+2
1030 FOR QQ=1 TO L/2
1040 PRINT LEFT$(A$,QQ):PRINT"[CRSRU]
";
1050 FOR KK=1 TO 90:NEXT KK
1060 FOR MM=1 TO L-1:PRINT"[CRSRR]";
:NEXT MM
1070 L=L-1
1080 PRINT RIGHT$(A$,QQ):PRINT"[CRSRU]
";
1090 FOR KK=1 TO 90:NEXT KK
1995 NEXT QQ
1998 RETURN
2000 REM SUBRUTINA PRINT NUMERO 2***
************************************
*******
2010 FOR MM=1 TO LEN(A$)
2020 IF MID$(A$,MM,1)="[SPC]" THEN V
=0:GOTO     2040
2030 V= 50
2040 FOR QQ= 1 TO 39
2050 PRINT MID$(A$,MM,1);
2060 FOR KK=1TO V:NEXT KK
2070 PRINT"[CRSRL][SPC]";
2080 NEXT QQ
2090 PRINT"[CRSRD]";
3000 FOR QQ=1 TO 39-MM
3050 PRINT MID$(A$,MM,1);
3060 FOR KK=1 TO V:NEXT KK
3070 PRINT"[CRSRL][SPC][2CRSRL]";
3080 NEXT QQ
3090 PRINT MID$(A$,MM,1)
4000 PRINT"[3CRSRU]";
4010 NEXT MM
4020 RETURN
5000 REM SUBRUTINA PRINT NUMERO 3***
************************************
*******
5010 L= LEN(A$):C= L*2
5020 PRINT TAB(L+2); A$
5030 FOR MM= L TO 1 STEP -1
5040 PRINT"[CRSRU]";TAB(C+1);"[SPC][CRSRU]
[CRSRL]";
5050 FOR QQ = L TO 0 STEP -1
5060 PRINT MID$(A$,MM,1);
5070 FOR KK=1 TO 70: NEXT KK
5080 PRINT"[CRSRL][SPC][2CRSRL]";
5090 NEXT QQ
6000 PRINT"[CRSRD]"; MID$(A$,MM,1):C
=C-1
6010 NEXT MM
6020 RETURN
```



Aplástalo y portadas

C-64

Román Estébanez Santesteban
16 años - S. Fermín, 49 - 1º izqda.
PAMPLONA (Navarra)
Tel. 236 372

Me llamo Román Estébanez y vivo en Pamplona. Poseo un commodere 64, un cassette y una impresora MPS-801. Os mando estos trucos y dos programas que espero sean de vuestro agrado.

Los trucos son los siguientes:

1. Este truco facilita el paso en la impresora del modo mayúsculas y gráficos al de minúsculas. Consiste en pulsar dentro del PRINT# la tecla del cursor hacia abajo. En la pantalla aparecerá un Q en vídeo inverso. Esto hará que la impresora comience a escribir en minúsculas sin necesidad de usar ningún CHR$.

Para volver al modo mayúsculas basta con pulsar dentro de las comillas la tecla del cursor hacia arriba.

2. Se trata de un truco para facilitar la búsqueda de un programa en el cassette. Basta con colocar un cable desde la patilla Nº 5 de la salida audio/vídeo al cable Nº 4-d (blanco-lectura del cassette) en el port del cassette.

Abriendo el volumen con un POKE 54296,1 podremos monitorear a través del altavoz de nuestro televisor lo que hay grabado en la cinta, ayudándonos a encontrar los programas por su pitido inicial.

NOTA: Este truco es muy sencillo en el cassette antiguo, ya que basta con introducir un cable pelado por el hueco que deja el cable blanco en la parte trasera del conector. En el modelo moderno habría que desmontar la clavija y hacer la conexión oportuna.

Aplástalo

El juego consiste en aplastar a todos los "malos" posibles en el menor tiempo. Tú conduces al último ejemplar de la raza humana, y en un desesperado intento, has de aniquilar a todos los enemigos que se crucen en tu camino. Dispones de dos minutos.

El procedimiento para aplastarlos es empujando unos enormes bloques de piedra de manera que quede aplastado por el enorme peso y perezca..., pero eso no es todo. Tu enemigo posee varias vidas y se reencarnará en otro punto de la pantalla hasta que se agoten sus "nuevas vidas". Por cada pantalla que logres superar, el número de reencarnaciones se incrementará, haciendo más difícil tu labor (este número se indica en la parte superior derecha de la pantalla).

Al término de cada pantalla se mostrará el tiempo invertido y una bonificación acorde con el tiempo.

Otro procedimiento de obtener puntos es juntando los tres valiosos diamantes dispersados por la pantalla. En un principio son de color amarillo, para que puedas divisarlos fácilmente, pero cuando los mueves retornan a su color negro. Si lo consigues lograrás 10.000 puntos.

En principio dispones de tres vidas, indicadas en la parte superior central de la pantalla. Si bates la puntuación récord, el ordenador te pedirá que entres tu nombre para figurar así en la tabla mundial de aplastadores. ¡Animo y suerte!

DESCRIPCION DEL PROGRAMA

Líneas	Comentario
10-160	Inicialización y generación de caracteres.
200-290	Inicialización pantalla de juego.

COMMODORE 64 LE DA ACCESO A MUCHOS ACCESORIOS

Unidad simple de disco (Monofloppy) 170 K.

Cassette.

Plotter e impresora, 4 colores, 14 c.p.s.

Impresora matricial, tractor, 30 c.p.s.

Interface RS232.

Joy Stick.

Paddle.

Cursos de Introducción al BASIC.



COMMODORE 64 LE MUESTRA PARTE DE SUS PROGRAMAS

Utilitarios y lenguajes

MONITOR LENGUAJE MAQUINA.
FORTH.
LOGO.
PILOT.
MACRO ASSEMBLER.
PROGRAMMER'S UTILITIES.
TURTLE GRAPHICS II.
MASTER.

Sistemas operativos

FILE/BOSS.
CP/M.

Programas de aplicaciones

EASY SCRIPT.
Proceso de texto de gran potencia.

CALC RESULT.
Hoja electrónica de cálculo.

EASY CALC RESULT.
Versión simplificada del CALC RESULT.

MAGIC DESK.
Proceso de texto y gestión de ficheros.

AGENDA TELEFONICA.

Programas educativos

MUSIC MACHINE.
MUSIC COMPOSER.
VISIBLE SOLAR SYSTEM.
SPEED/BINGO MATH.
FISICA I.
MATEMATICAS I.
HISTORIA I.
GEOGRAFIA I.
GEOGRAFIA II.
JUEGOS EDUCATIVOS.
TEMAS MONOGRAFICOS.
CONOCIMIENTOS GENERALES.
QUIMICA I.

Juegos

JUPITER LANDER.
KICKMAN.
SEAWOLF.
RADAR RAT RACE.
TOOTH INVADERS.
LAZARIAN.
OMEGA RACE.
LE MANS.
PINBALL SPECTACULAR.
AVENGER.
SUPERMASH.
FROGMASTER.
GRID RUNNER.
ATTACK OF THE MUTANT CAMELS.
THE PIT.
MR. TNT.
6 GAME PROGRAMS.
BINGO.
ROOTING TOOTING.
MINESSOTA FAT'S POOL CHALLENGE.

... y seguimos ampliando la lista

El ordenador personal de la familia más potente

commodore
COMPUTER

MICROELECTRONICA Y CONTROL, S.A.
c/. Taquígrafo Serra, 7, 5.°. Barcelona-29
c./ Princesa, 47, 3.° G. Madrid-8

300-540	Rutina de juego.
540-670	Controlador del movimiento del bloque de piedra.
700-730	Examina posición de los diamantes.
1000-1150	Presentación del juego.
2000-2030	Bonos extras por juntar los diamantes.
2500	Actualiza la puntuación.
3000-3080	Pantalla de tiempo realizado en el juego.
3500-3520	Actualiza marcador de supervivencia (Nº de vidas).
4000-4040	Rutina variable de sonido.
4100-4130	Sonido de muerte del humano.
4300-4320	Reencarnación del "malo".
5500-5510	Fin de tiempo.
6000	Rutina de inicialización del sonido.
6500	Cálculo de números aleatorios.
7000-7030	Localización de la reencarnación (posición en pantalla).
8000-8020	Pantalla de récord batido.
9500-9520	Datas del humano, malo y bloques.

Este programa usa el joystick en el pert. 2. Moviéndolo en las 4 direcciones te desplazas por la pantalla, y apretando simultáneamente el botón de disparo empujas al bloque en la dirección deseada.

Portadas

Este sencillo programa sirve para hacer portadas para los cassettes, de manera que sean fácilmente reconocibles.

Al ejecutarlo pide por pantalla el nombre del programa que va a ocupar la primera cara, luego el de la segunda y finalmente el nombre del autor o autores.

Automáticamente la impresora empieza a realizar la tapa, marcando los límites superiores, inferiores y lateral derecho. El izquierdo queda delimitado por el propio papel. El texto que imprime es susceptible de modificación. Yo opté por esta presentación por gusto particular.

```
5 REM      *****   APLASTALO   *****
10 QQ=53280:V=54272:POKEV+24,15:DIM
SP(15):H$="CBM[SPC]64":HI=20000
20 FOR R=1 TO 5:READ CO(R):NEXT R:FO
RR=1 TO 12:READ SP(R):NEXT R
40 D(1)=1:D(2)=-1:D(3)=40:D(4)=-40:S
S=1024:GC=12288:JO=56320
60 IF PEEK(GC)=56 THEN 160
70 POKE52,48:POKE56,48
80 POKE56334,PEEK(56334)AND 254:PRIN
T"[CLR][4CRSRD][4SPC][RVSON]GENERAND
O[SPC]CARACTERES"
90 POKE 1,PEEK(1)AND 251
100 FOR I=8 TO 1232:POKEGC+I,PEEK(I+
53248):NEXT
110 FOR T=216 TO 247:READ A:POKEGC+T
,A:NEXT T
120 FOR I=GC TO GC+7:READ A:POKEI,A:
NEXT I
130 FOR I=1280 TO 1287:READ A:POKEI+
GC,A:NEXT
135 FOR I=648 TO 655:READ A:POKEI+GC
,A:NEXT
140 POKE 1,PEEK(1)OR 4:POKE 56334,PE
EK(56334)OR1:
160 POKE53272,(PEEK(53272)AND240)+12
:POKE 53281,8:POKE 53280,6:GOTO 1000

200 PRINT"[CLR][2CRSRD][2SPC][COMMA]
[33SHIFTC][COMMS]"
220 FORT=1TO20:PRINT"[2SPC][BLK][SHIFTB]
[BLK][33SPC][BLK][SHIFTB]":NEXT T
230 PRINT"[2SPC][BLK][COMMZ][33SHIFTC]
[COMMX]"
231 PRINT"[SPC]HIGH[SPC]SCORE[SPC]:"
HI".....H$;:PRINT"[HOM][SPC]UP1[9SPC]
VIDAS:[6SPC]MALOS:"
237 GOSUB 3500:FORT=1TONB:GOSUB 6500

260 POKE SS+X1+40*Y1,160:NEXT:IR=0:F
R=250:SR=10:NV=1
270 FOR U=1 TO 3
275 X=INT(RND(0)*31)+4
280 Y=INT(RND(0)*18)+5:IF PEEK(SS+40
*Y)<>32 THEN 275
285 POKESS+X+40*Y,90:POKESS+X+40*Y+5
4272,7:GOSUB 4000:NEXTU
290 X=13:Y=10:TI$="000000":MM=64:PB=
100:S2=0
300 P=SS+X+40*Y:POKE P,0:Q=SS+IN:POK
EQ,81
310 RR=PEEK(Q+D(S)):IFRR=0 THEN 500
315 IF TI$>"000200" THEN 5500
320 IF RR <>32 THEN S=INT(RND(0)*4)+
1:GOTO 310
350 IN=IN+D(S):O=PEEK(JO):IF O=127 T
HEN FORR=1TO30:NEXTR:GOTO 440
360 IF (OAND8)=0 THEN X=X+1:N=1:CA=2
7:GOTO 410
370 IF (OAND4)=0 THEN X=X-1:N=-1:CA=
28:GOTO 410
380 IF (OAND2)=0 THEN Y=Y+1:N=40:CA=
30:GOTO 410
390 IF (OAND1)=0 THEN Y=Y-1:N=-40:CA
=29:GOTO 410
400 GOTO 440
410 IF(PEEK(P+N)=160ORPEEK(P+N)=90)A
ND(OAND16)=0THENXX=N:POKEP,CA:GOSUB6
00
420 POKEP,32:IF SW=0 THEN SW=1:POKEP
O-XX,42:FORR=1TO60:NEXT R:POKEPO-XX,
32
430 IF PEEK(P+N)<>32 THEN X=X-N:GOTO
 440
440 POKEQ,32:GOTO 300
500 POKEP,81:POKE Q,32:NV=7:GOSUB 41
00:VI=VI-1:IFVI<1THEN1000
540 GOTO 200
600 GOSUB 6000:POKE V+4,129
610 PO=P+XX:SW=0:PA=PEEK(PO):II=INT(
RND(0)*4)+5:YU=0:POKEPO+54272,0
620 PO=PO+XX:YU=YU+II:IFPEEK(PO)=81
THEN S1=1:MP=81:GOTO 640
630 IF PEEK(PO)<>32 THEN 8000
640 POKEV+1,YU:POKEPO,PA:POKE PC,CO:
POKE PO-XX,32:SW=1:GOTO 620
660 IF PEEK(PO-XX)=90 AND S2=0  THEN
 GOSUB 700
670 POKE V+4,0:RETURN
700 W=PO-XX:IF S2=1 THEN RETURN
710 FOR T=1 TO 12 STEP 2
720 IFPEEK(W+SP(T))=90ANDPEEK(W+SP(T
+1))=90THENFORR=0TO9:GOSUB2000:NEXT:
RETURN
730 NEXT T:RETURN
 1000 IF SC>HI THEN 9000
 1001 POKE V+24,15+64:POKE V+22,255:G
 OSUB 6000:POKE V+4,129:POKE V+1,100:
 NV=1
 1002 POKE V+23,1
 1003 FOR T=255 TO 10 STEP- .5:POKE V
 +1,T:POKE V+22,T:NEXT T
 1025 PRINT"[CLR][CRSRD][BLK][5SPC]RO
 MAN[SPC]ESTEBANEZ[SPC]PRESENTA...":I
 R=230:FR=10:SR=-20:GOSUB 4000
 1026 PRINT"[3CRSRD][10SPC]***[SPC]AP
 LASTALO[SPC]***":GOSUB 4000
 1030 PRINT"[2CRSRD][BLK][6SPC]][SPC]
 [[SPC]@[SPC]£[SPC]][SPC]...[SPC]ESTE
 [SPC]ERES[SPC]TU.":GOSUB 4000
 1040 PRINT"[CRSRD][3SPC][RED][SHIFTQ]
 [SPC][BLU][SHIFTQ][SPC][COMM6][SHIFTQ]
 [SPC][BLK]...[SPC]ESTOS[SPC]SON[SPC]
 TUS[SPC]ENEMIGOS":GOSUB 4000
 1050 PRINT"[2CRSRD][4SPC]DESPLAZA[SPC]
 LOS[SPC]BLOQUES[SPC]DE[SPC]PIEDRA":G
 OSUB 4000
 1060 PRINT"[2CRSRD][4SPC]PARA[SPC]AP
```

```
LASTAR[SPC]A[SPC]TUS[SPC]CONTRARIOS.
":GOSUB 4000
1070 PRINT"[4CRSRD][2SPC]**[SPC]PULS
AR[2SPC]´FUEGO´[2SPC]PARA[SPC]COMENZ
AR[SPC]**":GOSUB 4000
1080 IF PEEK(JO)=111 THEN 1090
1081 FOR T=SS+54486 TO SS+54495:POKE
T,CO:NEXT T:CO=CO+1.5:IFCO>30 THEN C
O=0
1082 GOTO 1080
1090 IN=172:S=2:S1=0:NB=140:BA=3:VI=
3:PP=20:SC=0
1150 GOTO 200
2000 PRINT"[HOM][CRSRD][4SPC]***[SPC]
10.000[5SPC]BONUS[SPC]POINTS[SPC]***
":POKE 53281,2:POKE 53280,2
2015 IR=200:FR=40:SR=-20:NV=30:GOSUB
 4000:S2=1:POKE53281,8:POKE53280,2
2030 PRINT"[HOM][CRSRD][35SPC]":SC=S
C+10000:GOSUB2500:RETURN
2500 PRINT"[HOM][4CRSRR]"SC:RETURN
3000 POKE 53280,2:PRINT"[CLR][10SPC]
CLASIFICACION":S2=0
3015 M$=MID$(TT$,3,2):R$=RIGHT$(TT$,
2)
3020 PRINT"[5CRSRD][7SPC]TIEMPO:":PR
INT"[CRSRD][7SPC]"M$"[SPC]MINUTOS
3040 PRINT"[CRSRD][7SPC]"R$"[SPC]SEG
UNDOS.":TP=0
3060 BP=(120-VAL(R$))*100:BM=0:IF M$
="00"THEN BP=BP*2:GOTO 3070
3070 PRINT"[4CRSRD][5SPC]TIME[SPC]PO
INTS="BP:SC=SC+BP:BM=0:NV=10:BA=BA+1
:GOSUB 4000
3075 IF BA>8 THEN BA=8
3080 FORR=1TO2000:NEXTR:POKE 53280,6
:GOTO 200
3500 PRINT"[HOM]"TAB(32)"[7SPC]";:GO
SUB 2500
3510 FOR T=1044 TO 1043+VI:POKET,0:N
EXT
3520 FOR T=1055 TO 1055+BA-BM:POKET,
81:POKET+54272,0:NEXT:RETURN
4000 GOSUB 6000:FOR R=1TONV:FOR T=IR
TOFRSTEPSR
4040 POKEV+1,T:NEXTT,R:POKEV+4,0:RET
URN
4100 GOSUB 6000
4120 FOR T=200 TO 0 STEP -10
4130 FOR R=T TO T+10:POKEV+1,R:NEXT
R,T:POKEV+4,0:RETURN
4300 POKE Q+54272,0:PB=(2↑BM)*100:IR
=5:FR=60:SR=5
4310 FORU=1 TO 5:POKE Q,CO(U):GOSUB
4000:NEXT U
4320 POKE Q+54272,0:RETURN
5500 PRINT"[HOM][9CRSRD][6CRSRR][SPC]
[3CRSRR]**[SPC]FIN[SPC]DEL[SPC]TIEMP
O[SPC]**"
5510 IR=255:FR=20:SR=-1:GOSUB 4000:G
OTO500
6000 POKEV+5,16:POKEV+6,128:POKEV,75
:POKEV+4,33:RETURN
6500 X1=INT(RND(0)*33)+3:Y1=INT(RND(
0)*20)+3:RETURN
7000 GOSUB 6500
7020 IN=Y1*40+X1:BM=BM+1:Q=SS+IN:SC=
SC+PB:NV=1
7025 IF BM=BA+1 THENTT$=TI$:FORT=1TO
130:POKEQQ,T:NEXT:NB=NB+5:MA=MA+.5:G
OTO 3000
7030 GOSUB4300:GOSUB3500:GOTO 670
8000 IF S1=1 THEN S1=0:GOSUB 7000
8010 GOTO 660:PRINT"[YEL][COMM1]":NV
=1:IR=200:FR=20:SR=-10:GOSUB 400
9000 PRINT"[CLR][4CRSRD][5SPC]HAS[SPC]
BATIDO[SPC]EL[SPC]RECORD[SPC]!!!"
9010 FORTY=1TO2:IR=10:FR=200:SR=10:G
OSUB4000:IR=200:FR=10:SR=-10:GOSUB40
00:NEXT
9050 PRINT"[3CRSRD][3SPC]TU[SPC]NOMB
RE[SPC]";:INPUT H$:IF LEN(N$)>12 THE
N PRINT"[5CRSRU]":GOTO 9050
9060 PRINT"[4CRSRD][2SPC]TE[SPC]TEND
RE[SPC]EN[SPC]CUENTA[SPC]PARA[SPC]LO
S":HI=SC
9070 PRINT"[2CRSRD][7SPC]CAMPEONATOS
[SPC]DEL[SPC]MUNDO.":FORR=1TO3000:NE
XTR:GOTO 1000
9502 DATA 46,43,42,87,81,1,2,1,-1,-1
,-2,-40,-80,40,-40,40,80
9505 DATA28,8,15,8,8,24,36,68,56,16,
240,16,16,24,36,34,186,146,124,16,16
,16,40
9510 DATA 108,0,0,0,56,84,146,170,17
0,56,16,124,84,84,16,40,102
9520 DATA 126,195,153,165,165,153,19
5,126,130,68,56,40,126,56,68,130
```

```
10 REM PORTADAS DE CASSETTES
20 REM
30 REM ROMAN ESTEBANEZ....22-05-1984

35 REM
50 PRINT"[CLR][2CRSRD][6SPC]PORTADAS
[SPC]DE[SPC]CASSETTE"
60 PRINT"[3CRSRD]NOMBRE[SPC]DEL[SPC]
PROGRAMA[SPC]CARA[SPC]1";:INPUT N1$
65 PRINT"[3CRSRD]NOMBRE[SPC]DEL[SPC]
PROGRAMA[SPC]CARA[SPC]2";:INPUT N2$
70 PRINT"[2CRSRD]NOMBRE[SPC]DEL[SPC]
AUTOR";:INPUT NA$:IF N2$="" THEN NP$
=N1$:GOTO 80
75 NP$=N1$+"[SPC]&[SPC]"+N2$:IF LEN(
NP$)>37 THEN RUN
80 OPEN 3,4:B$=CHR$(16)
85 GOSUB 200:PRINT#3:PRINT#3
90 PRINT#3,CHR$(14)"[CRSRD]**[SPC][SHIFTC]
OMMODORE[SPC]64[SPC]**[SPC][CRSRU][SHIFTB]
"
95 PRINT#3:PRINT#3,CHR$(15)"[CRSRD][11SPC]
---[SPC]SOFTWARE[SPC]---[SPC][CRSRU]
"B$"38[COMMN]"
100 PRINT#3:FORT=1TO(38-(LEN(NP$)+4)
)/2:PRINT#3,"[SPC]";:NEXT T
110 FOR T=1 TO LEN(NP$)+4:PRINT#3,"[SHIFTQ]
";:NEXT T:PRINT#3
115 FORT=1TO(38-(LEN(NP$)+4))/2:PRIN
T#3,"[SPC]";:NEXT T
120 PRINT#3,"[SHIFTQ][SPC]"NP$"[SPC]
[SHIFTQ]"B$"38[COMMN]"
125 FORT=1TO(38-(LEN(NP$)+4))/2:PRIN
T#3,"[SPC]";:NEXT T
130 FOR T=1 TO LEN(NP$)+4:PRINT#3,"[SHIFTQ]
";:NEXT T:PRINT#3
135 PRINT#3:PRINT#3,"[CRSRD][7SPC][SHIFTB]
Y[SPC]"NA$"[CRSRU]";B$"38[COMMN]":PR
INT#3
140 PRINT#3:PRINT#3,CHR$(14)"[CRSRD]
**[SPC][SHIFTC]OMMODORE[SPC]64[SPC]*
*[SPC][CRSRU][SHIFTB]"CHR$(15)
145 PRINT#3:GOSUB 200
150 PRINT#3,"[CRSRD][SPC][SHIFTC]ARA
[SPC]A:[SPC][CRSRU]"N1$
155 PRINT#3,"[CRSRD][SPC][SHIFTC]ARA
[SPC]B:[SPC][CRSRU]"N2$
160 GOSUB 200
165 PRINT#3:PRINT#3,"[CRSRD][7SPC][SHIFTB]
Y[SPC]"NA$"[CRSRU]"B$"38[COMMN]"
170 PRINT#3:PRINT#3:PRINT#3:GOSUB 20
0
180 RUN
200 FOR T=1 TO 39:PRINT#3,"-";:NEXT:
PRINT#3:RETURN
```

El Misterio de los Pokes (VI)

Por Diego ROMERO

He recibido bastantes sugerencias por teléfono y por correo sobre esta serie de artículos, y como toda la revista está abierta a opiniones constructivas, creo que tenéis razón en que es mejor publicar el mapa de memoria completo y luego ir analizando las distintas partes.

Algunas de las cartas recibidas decían que no estaba siendo suficientemente detallado con las funciones de las posiciones de memoria, por ejemplo con la posibilidad de copiar una zona de memoria en otra utilizando las posiciones de memoria 88-96 (publicado en magia en el número 6), la posibilidad de eliminar la interrogación en los inputs, etc.

Creo que en mi defensa podría mencionar las cartas recibidas en sentido contrario (querían que fuese más rápido).

Como para poder contentar a todos o al menos a la mayoría debería hacer una encuesta y atenerme a los resultados, pero esto tardaría bastante y muchas veces las encuestas no son fiables porque los que se suelen quejar no las contestan... pasamos a publicar el mapa de memoria del C-64, que fue publicado en Club Commodore, luego el del VIC, y quizás alguno de los PETs. Cuando terminemos con todos ellos, creo que podemos dedicar esta sección a comentar las posibilidades de las posiciones de memoria más útiles y populares sin tener que seguir el orden numérico como hasta ahora, mientras tanto estoy abierto a sugerencias (las prefiero por carta, por favor)...

Hex	Decimal	Description
0000	0	Chip directional register
0001	1	Chip I/O; memory & tape control
0003-0004	3-4	Float-Fixed vector
0005-0006	5-6	Fixed-Float vector
0007	7	Search character
0008	8	Scan-quotes flag
0009	9	TAB column save
000A	10	0=LOAD, 1=VERIFY
000B	11	Input buffer pointer/# subscrpt
000C	12	Default DIM flag
000D	13	Type: FF=string, 00=numeric
000E	14	Type: 80=integer, 00=floating point
000F	15	DATA scan/LIST quote/memry flag
0010	16	Subscript/FNx flag
0011	17	0=INPUT;$40=GET;$98=READ
0012	18	ATN sign/Comparison eval flag
0013	19	Current I/O prompt flag
0014-0015	20-21	Integer value
0016	22	Pointer: temporary strg stack
0017-0018	23-24	Last temp string vector
0019-0021	25-33	Stack for temporary strings
0022-0025	34-37	Utility pointer area
0026-002A	38-42	Product area for multiplication
002B-002C	43-44	Pointer: Start-of-Basic
002D-002E	45-46	Pointer: Start-of-Variables
002F-0030	47-48	Pointer: Start-of-Arrays
0031-0032	49-50	Pointer: End-of-Arrays
0033-0034	51-52	Pointer: String-storage(moving down)
0035-0036	53-54	Utility string pointer
0037-0038	55-56	Pointer: Limit-of-memory
0039-003A	57-58	Current Basic line number
003B-003C	59-60	Previous Basic line number
003D-003E	61-62	Pointer: Basic statement for CONT
003F-0040	63-64	Current DATA line number
0041-0042	65-66	Current DATA address
0043-0044	67-68	Input vector
0045-0046	69-70	Current variable name
0047-0048	71-72	Current variable address
0049-004A	73-74	Variable pointer for FOR/NEXT
004B-004C	75-76	Y-save; op-save; Basic pointer save
004D	77	Comparison symbol accumulator
004E-0053	78-83	Misc work area, pointers, etc
0054-0056	84-86	Jump vector for functions
0057-0060	87-96	Misc numeric work area
0061	97	Accum#1: Exponent
0062-0065	98-101	Accum#1: Mantissa
0066	102	Accum#1: Sign
0067	103	Series evaluation constant pointer
0068	104	Accum#1 hi-order (overflow)
0069-006E	105-110	Accum#2: Exponent, etc.
006F	111	Sign comparison, Acc#1 vs #2
0070	112	Accum#1 lo-order (rounding)
0071-0072	113-114	Cassette buff len/Series pointer
0073-008A	115-138	CHRGET subroutine; get Basic char
007A-007B	122-123	Basic pointer (within subrtn)
008B-008F	139-143	RND seed value
0090	144	Status word ST
0091	145	Keyswitch PIA: STOP and RVS flags
0092	146	Timing constant for tape
0093	147	Load=0, Verify=1
0094	148	Serial output: deferred char flag
0095	149	Serial deferred character
0096	150	Tape EOT received
0097	151	Register save
0098	152	How many open files
0099	153	Input device, normally 0
009A	154	Output CMD device, normally 3
009B	155	Tape character parity
009C	156	Byte-received flag
009D	157	Direct=$80/RUN=0 output control
009E	158	Tp Pass 1 error log/char buffer
009F	159	Tp Pass 2 err log corrected
00A0-00A2	160-162	Jiffy Clock HML
00A3	163	Serial bit count/EOI flag
00A4	164	Cycle count
00A5	165	Countdown,tape write/bit count
00A6	166	Tape buffer pointer
00A7	167	Tp Wrt ldr count/Rd pass/inbit
00A8	168	Tp Wrt new byte/Rd error/inbit cnt
00A9	169	Wrt start bit/Rd bit err/stbit
00AA	170	Tp Scan;Cnt;Ld;End/byte assy
00AB	171	Wr lead length/Rd checksum/parity
00AC-00AD	172-173	Pointer: tape bufr, scrolling
00AE-00AF	174-175	Tape end adds/End of program
00B0-00B1	176-177	Tape timing constants
00B2-00B3	178-179	Pntr: start of tape buffer
00B4	180	1=Tp timer enabled; bit count
00B5	181	Tp EOT/RS232 next bit to send
00B6	182	Read character error/outbyte buf
00B7	183	# characters in file name
00B8	184	Current logical file
00B9	185	Current secndy address
00BA	186	Current device
00BB-00BC	187-188	Pointer to file name
00BD	189	Wr shift word/Rd input char
00BE	190	# blocks remaining to Wr/Rd
00BF	191	Serial word buffer
00C0	192	Tape motor interlock
00C1-00C2	193-194	I/O start address

Hex	Decimal	Description
00C3-00C4	195-196	Kernel setup pointer
00C5	197	Last key pressed
00C6	198	# chars in keybd buffer
00C7	199	Screen reverse flag
00C8	200	End-of-line for input pointer
00C9-00CA	201-202	Input cursor log (row, column)
00CB	203	Which key: 64 if no key
00CC	204	0=flash cursor
00CD	205	Cursor timing countdown
00CE	206	Character under cursor
00CF	207	Cursor in blink phase
00D0	208	Input from screen/from keyboard
00D1-00D2	209-210	Pointer to screen line
00D3	211	Position of cursor on above line
00D4	212	0=direct cursor, else programmed
00D5	213	Current screen line length
00D6	214	Row where curosr lives
00D7	215	Last inkey/checksum/buffer
00D8	216	# of INSERTs outstanding
00D9-00F2	217-242	Screen line link table
00F3-00F4	243-244	Screen color pointer
00F5-00F6	245-246	Keyboard pointer
00F7-00F8	247-248	RS-232 Rcv pntr
00F9-00FA	249-250	RS-232 Tx pntr
00FF-010A	256-266	Floating to ASCII work area
0100-103E	256-318	Tape error log
0100-01FF	256-511	Processor stack area
0200-0258	512-600	Basic input buffer
0259-0262	601-610	Logical file table
0263-026C	611-620	Device # table
026D-0276	621-630	Sec Adds table
0277-0280	631-640	Keybd buffer
0281-0282	641-642	Start of Basic Memory
0283-0284	643-644	Top of Basic Memory
0285	645	Serial bus timeout flag
0286	646	Current color code
0287	647	Color under cursor
0288	648	Screen memory page
0289	649	Max size of keybd buffer
028A	650	Repeat all keys
028B	651	Repeat speed counter
028C	652	Repeat delay counter
028D	653	Keyboard Shift/Control flag
028E	654	Last shift pattern
028F-0290	655-656	Keyboard table setup pointer
0291	657	Keyboard shift mode
0292	658	0=scroll enable
0293	659	RS-232 control reg
0294	660	RS-232 command reg
0295-0296	661-662	Bit timing
0297	663	RS-232 status
0298	664	# bits to send
0299-029A	665	RS-232 speed/code
029B	667	RS232 receive pointer
029C	668	RS232 input pointer
029D	669	RS232 transmit pointer
029E	670	RS232 output pointer
029F-02A0	671-672	IRQ save during tape I/O
02A1	673	CIA 2 (NMI) Interrupt Control
02A2	674	CIA 1 Timer A control log
02A3	675	CIA 1 Interrupt Log
02A4	676	CIA 1 Timer A enabled flag
02A5	677	Screen row marker
02C0-02FE	704-766	(Sprite 11)
0300-0301	768-769	Error message link
0302-0303	770-771	Basic warm start link
0304-0305	772-773	Crunch Basic tokens link
0306-0307	774-775	Print tokens link
0308-0309	776-777	Start new Basic code link
030A-030B	778-779	Get arithmetic element link
030C	780	SYS A-reg save
030D	781	SYS X-reg save
030E	782	SYS Y-reg save
030F	783	SYS status reg save
0310-0312	784-785	USR function jump (B248)
0314-0315	788-789	Hardware interrupt vector (EA31)
0316-0317	790-791	Break interrupt vector (FE66)
0318-0319	792-793	NMI interrupt vector (FE47)
031A-031B	794-795	OPEN vector (F34A)
031C-031D	796-797	CLOSE vector (F291)
031E-031F	798-799	Set-input vector (F20E)
0320-0321	800-801	Set-output vector (F250)
0322-0323	802-803	Restore I/O vector (F333)
0324-0325	804-805	INPUT vector (F157)
0326-0327	806-807	Output vector (F1CA)
0328-0329	808-809	Test-STOP vector (F6ED)
032A-032B	810-811	GET vector (F13E)
032C-032D	812-813	Abort I/O vector (F32F)
032E-032F	814-815	Warm start vector (FE66)
0330-0331	816-817	LOAD link (F4A5)
0332-0333	818-819	SAVE link (F5ED)
033C-03FB	828-1019	Cassette buffer
0340-037E	832-894	(Sprite 13)
0380-03BE	896-958	(Sprite 14)
03C0-03FE	960-1022	(Sprite 15)
0400-07FF	1024-2047	Screen memory
0800-9FFF	2048-40959	Basic ROM memory
8000-9FFF	32768-40959	Alternate: ROM plug-in area
A000-BFFF	40960-49151	ROM: Basic
A000-BFFF	49060-59151	Alternate: RAM
C000-CFFF	49152-53247	RAM memory, including alternate
D000-D02E	53248-53294	Video Chip (6566)
D400-D41C	54272-54300	Sound Chip (6581 SID)
D800-DBFF	55296-56319	Color nybble memory
DC00-DC0F	56320-56335	Interface chip 1, IRQ (6526 CIA)
DD00-DD0F	56576-56591	Interface chip 2, NMI (6526 CIA)
D000-DFFF	53248-53294	Alternate: Character set

Hex	Decimal	Description
E000-FFFF	57344-65535	ROM: Operating System
E000-FFFF	57344-65535	Alternate: RAM
FF81-FFF5	65409-65525	Jump Table, Including:

FFC6 - Set Input channel
FFC9 - Set Output channel
FFCC - Restore default I/O channels
FFCF - INPUT
FFD2 - PRINT
FFE1 - Test Stop key
FFE4 - GET



A000; ROM control vectors
A00C; Keyword action vectors
A052; Function vectors
A080; Operator vectors
A09E; Keywords
A19E; Error messages
A328; Error message vectors
A365; Misc messages
A38A; Scan stack for FOR/GOSUB
A3B8; Move memory
A3FB; Check stack depth
A408; Cneck memory space
A435; "out of memory"
A437; Error routine
A469; BREAK entry
A474; "ready."
A480; Ready for Basic
A49C; Handle new line
A533; Re-chain lines
A560; Receive input line
A579; Crunch tokens
A613; Find Basic line
A642; Perform [NEW]
A65E; Perform [CLR]
A68E; Back up text pointer
A69C; Perform [LIST]
A742; Perform [FOR]
A7ED; Execute statement
A81D; Perform [RESTORE]
A82C; Break
A82F; Perform [STOP]
A831; Perform [END]
A857; Perform [CONT]
A871; Perform [RUN]
A883; Perform [GOSUB]
A8A0; Perform [GOTO]
A8D2; Perform [RETURN]
A8F8; Perform [DATA]
A906; Scan for next statement
A928; Perform [IF]
A93B; Perform [REM]
A94B; Perform [ON]
A96B; Get fixed point number
A9A5; Perform [LET]
AA80; Perform [PRINT#]
AA86; Perform [CMD]
AAA0; Perform [PRINT]
AB1E; Print string from (y.a)
AB3B; Print format character
AB4D; Bad input routine
AB7B; Perform [GET]
ABA5; Perform [INPUT#]
ABBF; Perform [INPUT]
ABF9; Prompt & input
AC06; Perform [READ]
ACFC; Input error messages

(continuará)

Magia



La MAGIA son trucos, la MAGIA es divertida.

La MAGIA es hacer lo que nadie se ha atrevido y resulta ser la fuente más completa de información para la informática práctica.

La MAGIA es una sección mensual llena de consejos, trucos, de esto y aquello del mundo del software, hardware y aplicaciones, trucos descubiertos por los demás que hacen que la informática sea más fácil, más divertida o más animada.

MAGIA habla de ideas sencillas, programas de una sola línea, subrutinas útiles, hechos de informática poco conocidos y otras cosas de interés.

Los trucos de magia enviárnoslos comprobados, pues hay varios incorrectos. ¡Ah! y no nos mandéis trucos repetidos... ¡Listos!. Premios: Ver pág. 5.

INICIACION AL TELEPROCESO

Dada la gran cantidad de usuarios de los Microordenadores aplicados a la radio RTTY/CW, os envío un corto pero eficaz programa para iniciaros en la Modalidad ASCII, con el cual podréis transmitir y recibir todos los caracteres Ascii de vuestro Ordenador, o sea que todo lo que hagáis vosotros en vuestra pantalla le saldrá al correspondal, a diferencia del código BAUDOT que queda limitado a emitir letras números y poco más.

Con este programa que es el primero de una serie que posteriormente publicaré queda la puerta abierta para el teleproceso:

Envio y recepción de listados.

Acceso a programas de otros ordenadores.

En fin todo cuanto os quiera dar la fantasía de los ordenadores aplicados a la radio.

Como es lógico para que el programa funcione al 100×100 el corresponsal tiene que tener otro ordenador igual sobre todo en lo que respecta a color etc., no sé cómo funcionaría teniendo el corresponsal otro ordenador distinto, yo las pruebas las he realizado con ordenadores iguales.

La eficacia en la transmisión también depende mucho del Modem a utilizar, yo las pruebas las he realizado con un Modem autoconstruido por mí y el resultado ha sido bastante satisfactorio incluso a 110 Baudios con señales en FM menores de UN Santiago de Smiter.

¡Animo y a darle a las teclas!

Arturo Sánchez Galán
Solosancho (Avila)
EA 1 BDD

PROGRAMA EMISION RECEPCION "ASCII"

```
10 OPEN 2,2,3, CHR$ (129)+CHR$(192)
20 PRINT"(clr) EMISION"
25 GET A$: IF A$=" ↑"THEN 100
30 PRINT A$;
40 PRINT #2,A$;
50 GOTO 25
100 PRINT "(clr) RECEPCION"
110 GET A$: IF A$ ="↑" THEN 20
120 GET#2,A$: PRINT A$;
130 GOTO 110
```

La velocidad en baudios puede aumentarse sumando uno al primer CHR$

50 Baudios = CHR$ (129)
75 Baudios = CHR$ (130)
110 Baudios = CHR$ (131)

No es recomendable pasar de una velocidad de 110 baudios a no ser que las señales de la otra estación sean muy limpias.

MERGE-64

Cuando en el C-64 se efectúa un LOAD automáticamente se borra el programa que hasta ese momento estuviese en la memoria RAM. Esto impide enlazar dos o más programas previamente almacenados en cinta. En otros ordenadores se dispone de la instrucción MERGE que permite realizarlo. Los usuarios del C-64 pueden simular esta instrucción de la siguiente forma:

—Hacer un LOAD del primer programa.

—Mover el puntero de inicio del BASIC de forma que apunte al final del programa que acabamos de cargar, para ello teclear:

POKE 43, PEEK(45) - 2: POKE 44, PEEK(46)

—Hacer un LOAD del segundo programa.

—Volver a colocar el puntero de inicio del BASIC en la posición $ 0801, para ello:

POKE 43, 1: POKE 44,8

Si después de todo esto hacéis un LIST, veréis que los dos programas aparecen enlazados: el programa que se cargó en 2º lugar comienza al final del primero.

Conviene que el programa que se vaya a cargar en 2º lugar tenga números de líneas superiores a los del programa que se encuentre en memoria ya que el orden de los números de línea no se restablece (ojo con los GOTO y GOSUB!)"

Rafael A. García Segura
Montes de Oca, 9 - 2º D
29007 MALAGA

MERGE VIC-20

Os envío una subrutina que algunos ordenadores poseen pero no el VIC-20, se trata de la función **"MERGE"**.

La función "MERGE" sirve para unir dos programas en uno solo. El procedimiento es el siguiente: Se comprueba primero que entre los dos programas a unir no exista ningún número de línea idéntico, ya que en este caso la línea se perdería al ser el número igual. Luego se teclea en modo directo POKE 43, PEEK (45)-2: POKE 44, PEEK (46), y se carga el segundo programa con LOAD. Cuando éste se haya cargado se teclea en modo directo POKE 43,1: POKE 44,16, y ya se pueden listar los dos programas unidos.

Andres Portas Afenjar
C/Ríos Rosas, 5-4º D
29007 MALAGA

QUINIELAS

Poseo un CBM 64 desde hace 4 meses, y desde entonces estoy suscrito a vuestra revista Nº 3332.

Hace muy pocos días adquirí la impresora MP8 80V, y os mando un pequeño programa para rellenar quinielas con esta impresora.

Os mando el listado y un ejemplo práctico en papel, de la impresora. Por supuesto es totalmente aleatorio, pero podría ser la de la próxima semana (por si acaso copiarla).

```
READY.

5 REM ** QUINIELA **
7 REM ** JOSE A. FERNANDEZ VILLORIA."
10 OPEN1,4
20 PRINT#1,CHR$(14)CHR$(16)"25QUINIELA"
21 PRINT#1,CHR$(18)CHR$(16)"25          "
25 PRINT#1,CHR$(15)
30 FORH=1TO60:PRINT#1,"-";:NEXTH
35 PRINT#1,CHR$(10)
40 FORI=1TO14
42 IFI<10THENPRINT#1," ";
45 PRINT#1,I;" PARTIDO";
50 FORJ=1TO8
60 W=INT(RND(0)*11):IFW<=5THENS$=" 1   "
70 IFW<=8ANDW>5THENS$="  X  "
80 IFW>8THENS$="   2 "
90 PRINT#1,CHR$(16)3S$;
110 NEXTJ
111 PRINT#1
112 NEXTI
113 PRINT#1,CHR$(16)"65 VILLORIA."
114 FORH=1TO60:PRINT#1,"-";:NEXTH
115 PRINT#1
120 CLOSE1

READY.
```

```
                        QUINIELA
------------------------------------------------------------

 1  PARTIDO    2   1       1       1      1      X     1     1
 2  PARTIDO   X    1        X      1      1      X     1     1
 3  PARTIDO    2   1       1        X      X     X     1     1
 4  PARTIDO   X      2       2     1        2     2    1     1
 5  PARTIDO   X      X      X      1      1      X     1      2
 6  PARTIDO    2     X     1         2    1      X      2     2
 7  PARTIDO  1       X     1         2    1      X     1     1
 8  PARTIDO   X      X       2       2      2     2    1     1
 9  PARTIDO   X    1       1       1       X     X     1     1
10  PARTIDO  1       X      X      1      1     1       X     2
11  PARTIDO    2     X       2     1      1     1      1     1
12  PARTIDO   X    1        X        2    1     1      1      X
13  PARTIDO  1     1       1         2     X    1      1      2
14  PARTIDO  1     1       1       1       X    1      1      2
------------------------------------------------------------
```

José Antonio Fernández Villoria
C/ García nº 5, 2º Izda.
Gijón (Asturias)

SALVAGUARDA DE MEMORIA

Queridos amigos de COMMODORE WORLD; es esta la primera vez que envío una colaboración a la revista, aunque soy suscriptor desde el nº 1, a ver si entro en el tinglado de los premios y regalos, ya que observo que para los sucriptores que se "quedan en casa" no hay nada.

Bien, paso a comentar mi "trabajillo":

Me voy a referir a un artículo que con el título "SALVAGUARDA DE MEMORIA EN PERIFERICO", apareció en la revista nº 6, firmado por Rafael Jiménez Jiménez, ya que es un programa interesante para todos aquellos que les guste programar en código máquina, toda vez que permite salvar las rutinas de una forma más rápida y ortodoxa que el encerrarlas en BASIC, amén de la facilidad de carga, ya que aunque tengamos otro programa en memoria, podemos cargar la rutina, sin afectar al programa existente, lo que no sería posible si la rutina estuviera en BASIC.

Pues bien, mis comentarios son los siguientes:

1º.—El programa que nos presenta nuestro amigo Rafael no funciona correctamente, ya que no salva el contenido de la dirección dada como dirección final a la pregunta que hace el programa. Se puede comprobar salvando una zona de memoria, se borra posteriormente, y se carga lo salvado, se observa que falta el contenido de la última dirección. La causa está en que la rutina SAVE del Kernal, no salva el contenido de la dirección especificada como final. El defecto del programa es importante, toda vez que las rutinas en código máquina, suelen terminar con la instrucción RETURN, y si esta falta, la rutina no va a funcionar, corriendo el peligro de causar un "atasco" al ordenador al lanzar la rutina. El problema se soluciona sustituyendo la línea 70 por la siguiente:

70 PRINT: INPUT"DIR. FINAL DE MEMORIA"; DY: DY=DY+1: GOSUB3000: LF=BB: MF=BA.

Con esto el problema queda solucionado, ya que el programa suma 1 a la dirección final.

2º.—Cuando una rutina salvada con este programa, se carga en modo directo, dicha carga afecta a ciertos punteros de página cero, de tal manera que si en este momento tenemos un programa BASIC en memoria, existen pocas posibilidades de que después de la carga de la rutina funcione, a menos que tomemos algunas precauciones antes de cargar la rutina. La causa de esto, tenemos que buscarla de nuevo en las rutinas del Kernal, en este caso en la rutina LOAD. Esta rutina, devuelve la última dirección cargada, dirección que utiliza el INTERPRETE BASIC para saber dónde termina el programa, y dónde por lo tanto puede almacenar las variables, para ello, almacena esta dirección devuelta por la rutina LOAD en los punteros de página cero $2D y $2E. Si la rutina cargada reside fuera de la zona de memoria BASIC, lo cual es bastante frecuente ya que la zona de memoria normalmente utilizada para las rutinas en código máquina esta entre 49152 y 53247, muy por encima de la zona BASIC, los punteros de la página cero se actualizan a un valor incorrecto, lo que ocasiona que el INTERPRETE BASIC emita el mensaje "OUT OF MEMORY", no permitiendo hacer nada a menos que se haga NEW, con lo que perdemos el programa.

Existen dos soluciones para este problema, la primera consiste en salvar los punteros de página cero, a continuación cargar la rutina, y posteriormente restaurar los punteros de nuevo; la segunda consiste en cargar la rutina con un LOAD en modo programa, en cuyo caso el INTERPRETE BASIC, no actualiza los punteros. A continuación se describen los dos métodos:

PRIMERA SOLUCION.—Cómo salvar y restaurar los punteros.

1º.—Hacer POKE1020,PEEK (45):POKE1021,PEEK (46) (RETURN)

2º.—Hacer LOAD"NOMBRE", NUMERO DE PERIFERICO,1 (RETURN)

3º.—Esperar que se cargue la rutina.

4º.—Hacer POKE45,PEEK(1020):POKE46,PEEK(1021) (RETURN)

La rutina se habrá cargado sin afectar al programa BASIC

SEGUNDA SOLUCION.—Carga de la rutina en modo programa.

Es necesario para ello añadir un par de líneas al programa BASIC de la siguiente forma:

1 END

2 LOAD "NOMBRE", NUMERO DE PERIFERICO, 1

Hacer RUN 2 (RETURN), y esperar a que se cargue la rutina, entonces borrar las líneas 1 y 2, con lo que tendremos cargada la rutina, y el programa BASIC correcto.

Y ya, para finalizar, ahí está mi propia versión del programa. No se utilizan las rutinas preparatorias SETNAM y SETLFS, ya que lo único que estas rutinas hacen es colocar datos en la página cero para su posterior utilización por la rutina SAVE. En este programa, el trabajo de esas rutinas lo realiza el BASIC directamente, con lo que el programa se hace mucho más corto, aunque es necesario entrar con un SYS en la dirección adecuada de la rutina SAVE; esto se hace al final de la línea 60. El programa comprueba también que la grabación ha sido correcta.

```
5 REM  *** MEMORIA A PERIFERICO ***
6 REM
10 PRINT"[CLR][2CRSRD]":INPUT"NOMBRE
[SPC]";N$:N=LEN(N$):IFN>10THEN N$=LE
FT$(N$,10):N=10
20 INPUT"[CRSRD]DIR.[SPC]INICIAL[SPC]
";D:POKE194,INT(D/256):POKE193,D-INT
(D/256)*256
30 INPUT"[CRSRD]DIR.[SPC]FINAL[SPC]"
;D:D=D+1:POKE175,INT(D/256):POKE174,
D-INT(D/256)*256
40 INPUT"[CRSRD]1-CASS.[SPC]8-DISCO"
;A:IFA<>1ANDA<>8GOTO40
50 POKE186,A:POKE183,N:POKE184,1:POK
E185,1:POKE187,192:POKE188,2
60 FORJ=1TO N:POKE703+J,ASC(MID$(N$,
J,1)):NEXT:SYS62957
70 PRINT"[2CRSRD]REBOBINA[SPC]LA[SPC]
CINTA[SPC]Y[SPC]PULSA[SPC]UNA[SPC]TE
CLA":POKE198,0:WAIT198,1
80 PRINT"[CLR]VERIFY"CHR$(34)N$CHR$(
34)","A",1"
90 POKE198,2:POKE631,19:POKE632,13:E
ND
```

Por último, aunque la sintaxis correcta de carga es:

LOAD "NOMBRE", NUMERO DE PERIFERICO, 1

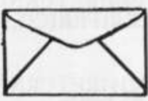
Si se utiliza como periférico el cassette, es suficiente hacer LOAD, sin ningún otro dato para la carga de las rutinas.

Alfredo Pérez Pérez
Bolarque - GUADALAJARA

CARTA BLANCA



TECLAS EXTRAÑAS



Me gustaría que me contestaran a un par de dudas que tengo sobre mi VIC-20. En primer lugar quisiera saber cuál es la función de tres teclas de la consola del ordenador, que son las siguientes:

"←" /// "£" /// "@"

EMILIO GRANES FERNANDEZ
C/PLAZA DE LA CARRETERIA, Nº 33
CONSTANTINA (SEVILLA)

Las teclas "←", "£" y "@" no tienen otra función más que la de servir de símbolos gráficos, aunque algunas de ellas, en determinados programas (ej. "WEDGE" con la unidad de discos) tengan una función especial.

GRAFICOS EN ROM



Lo primero felicitaros por vuestra revista, especialmente por la sección de magia que tantos trucos útiles aporta.

Mi consulta va sobre los caracteres programables.

Tras haber leído la sección del manual dedicada a esto, tantas veces como para gastar las letras, he conseguido enterarme de cómo funciona el programa de ejemplo para copiar caracteres de la ROM a la RAM.

Pero ahora que ya estaba contento me he encontrado con el problema (para mí hasta ahora insalvable) de que no sé (ni lo dice el manual) cómo copiar los caracteres del set 2 es decir mayúsculas y minúsculas.

Otra duda sobre la cual no he encontrado respuesta en ninguna revista o libro es de, qué hay que hacer para que un programa que se esté cargando desde el cassette o la unidad de discos se autoejecute.

Me despido no sin antes felicitaros nuevamente por vuestra/nuestra revista.

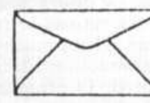
Un asiduo lector,
MIGUEL F. BAEZ DE AGUILAR G.
(MALAGA)

Las minúsculas, inversas, etc., están situadas en las ROM después de los caracteres en mayúsculas:

53248	*$D000*	*Mayúsculas*
53760	*$D200*	*Gráficos.*
54272	*$D400*	*Mayúsc. invertidas.*
54784	*$D600*	*Gráficos invertidos.*
55296	*$D800*	*Minúsculas.*
55808	*$DA00*	*Mayúsculas y gráfic.*
56320	*$DC00*	*Minúsculas inv.*
56832	*$DE00*	*Minúsculas y gráf. inv.*

A lo del auto-run, lo publicaremos en próximos números pues hemos recibido numerosas cartas al respecto.

COMPATIBILIDAD DE CINTAS



Hola, os escribo porque tengo un problema con mi VIC-20 y si podéis, quiero que me lo resolváis.

Tengo la posibilidad de intercambiar programas con VIC-iosos Ingleses, pero he oído que sus sistemas de televisión son PAL (o NTSC), y el nuestro es el de 625 líneas. Funciona una cinta con programas ingleses en un VIC español y viceversa.

JAVIER RECIO MATA
C/FERNAN CABALLERO, 20
41001 SEVILLA

En primer lugar la contestación rápida a tus dudas, las cintas utilizan el mismo sistema en todos los VIC-20 del mundo. En segundo lugar te indicaré que el sistema Inglés es el PAL, que es el mismo que el nuestro. La diferencia entre sus VIC-20's y los nuestros es que el canal de sonido del sistema PAL británico utiliza una subportadora de audio de frecuencia superior a la nuestra (6.0 MHZ en lugar de 5,5 MHz), esa es la causa de que los ordenadores traídos de Inglaterra no se oigan bien en nuestros televisores, y si ajustas la sintonía para oírlos bien entonces no lo ves. La solución es un ajuste que tiene el modulador (esa cajita que se enchufa al televisor). El caso es el mismo con los C-64 de las Canarias y que son extranjeros (hay que retocar un tornillito). Algunas veces hemos oído que tenían que cambiarles un carísimo circuito integrado, pero esta es una idea incorrecta.

GOSUBS ENCADENADOS

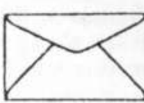


Poseo un VIC-20 desde primeros de este año y considero su labor muy interesante. Desde el punto de vista de un VIC-ioso su revista lleva corto el contenido para este ordenador aunque comprendo que no se puede abandonar a los **pobres** propietarios de C-64, 4000, 8000, etc.

En el MANUAL DEL USUARIO se comenta que después de un GOSUB, la instrucción RETURN devuelve el programa a la instrucción siguiente al GOSUB, pero ¿qué ocurre cuando hay dos GOSUBs encadenados? es lo mismo que con las instrucciones FOR ...NEXT?

Efectivamente, los GOSUBS pueden encadenarse como si fuesen FOR...NEXT, pueden llegar a encadenarse hasta 24 subrutinas unas dentro de otras; si se sobrepasa este número, el ordenador contesta con ?out of memory. Para comprobarlo basta con escribir... 1 GOSUB 1 y hacer RUN.

SOBRE LA ROM



Quisiera hacer unas cuantas preguntas, de las cuales no he encontrado respuesta en ningún sitio.

a) Al copiar la ROM en la RAM ¿hay algún método para copiar el intérprete del BASIC en cinta de cassette?

b) Este programa, por lo tanto, podría hacerse correr en otro microordenador con la misma CPU?

c) Si así fuera ¿este otro ordenador se podría hacer funcionar con los mismos programas que el primero?

d) Y por último, los cartuchos enchufables. ¿Sustituyen a la ROM? y si es así ¿ocupa las mismas posiciones de memoria?, ¿estos se pueden copiar en la RAM?

Os habré dejado aturdidos ¿no es así? Se despide de vosotros rápidamente:

IVO PLANA VALLVE
C/CAPUTXINS, 30, 4, 1ª
TARRAGONA

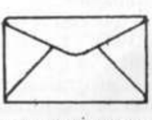
A.—Puedes salvarlo con un monitor de lenguaje máquina o con el programa aparecido en el número 6 (pág. 46) "Salvaguarda de memoria en periféricos".

B.—Sí, pero el mapa de memoria de este otro microordenador tiene que ser igual que la del C-64, es decir, o es un C-64 o una copia idéntica.

C.—Claro, sería un C-64 o su "fotocopia".

D.—Algunos la sustituyen, otros se suman a ella sin sustituirla, etc. Todo depende de la combinación de las señales EXRON, GAME, en el Port de expansión y el contenido de la dirección 1 (la tienes en los cuadros de "El Misterio de los Pokes" del número 4).

COMPRA DE PROGRAMAS

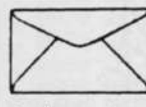


He comprado el número 5 de vuestra revista para informarme mejor sobre las posibilidades de el ordenador personal Commodore-64 puesto que pienso adquirir en breve uno. Al leer vuestra revista he visto con asombro la calidad y claridad de esta, que tratando temas con bastante profundidad se hace entender por personas que, como yo, no tenemos un nivel de conocimientos de informática demasiado elevado. También he leído algo que me ha llamado poderosamente la atención y es un boletín llamado "Servicio de Cintas", en el que, según creo entender, ofrece un servicio de venta en cintas magnetofónicas de programas "publicados" al precio único de 850 pesetas más 75 por gastos de envío. Es este el punto que os pido me aclaréis, el de los programas "publicados", o sea, si se puede comprar cualquier programa, ya sea de juegos o de gestión, por muy poco nombrado que esté en la revista, al precio antes mencionado.

MIGUEL NUÑEZ PEREYRA
LANZAROTE

Se puede comprar cualquier programa de los que aparecen listados en la revista, no los programas a los que hace referencia la publicidad.

FALLOS EN LOS PROGRAMAS



En primer lugar, quería felicitaros por vuestra revista, quizás un poco técnica, pero muy completa. Tengo un C-64, y acabo de adquirir el cassette, me ha ocurrido, que al copiar algunos programas, he encontrado fallos, que no he podido resolver, ni aún mirando las meteduras de pata de los siguientes números.

Los fallos son los siguientes:

—En el nº 3, dando valor a las teclas de funciones, el ordenador me da un Illegal quantity en la línea 1030.

—En el mismo nº 3, el programa "Mosqueando al mosquito", me da un error también, debido a que algún POKE maldito me ha borrado todos los signos más, menos, comillas, comas, etc... y además no puedo volver a escribirlos.

—Por último, en el manual, el programa de "teclado de piano" que hay al final, me da Illegal quantity en la línea 310. Tanto este error, como el de las teclas de funciones, son debidos a una secuencia ASC, la cual, aún no he podido descifrar.

ENRIQUE LAHUERTA PEREZ
C/HERMANOS IBARRA, 6-8
HOTEL PATRIA,
50001 ZARAGOZA

1.—El error puede que no esté en esa línea, sino en alguna otra, en la que se definen (F$(I) o J con unos valores que hacen ilegal su utilización en la línea 1030.

2.—Si haces RUN sin haber salvado el programa antes, te arriesgas a que algún POKE rarillo te bloquee el ordenador o te haga cosas que no debiera; si esto sucede, desconectar el ordenador y vuelves a cargar el programa que habías salvado.

3.—Pasa lo mismo que con la primera pregunta, en este caso verifica la línea 300.

INVERSION EN DATAS



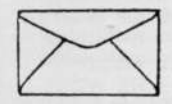
Me gustaría saber si se pueden leer los datos de las sentencias DATA de un programa directamente en orden inverso al que tienen.

Insisto en lo de "directamente", ya que el problema en sí tiene fácil solución metiendo previamente los datos en una matriz o haciendo un RESTORE y leyendo en cada vuelta un dato menos.

FRANCISCO SAEZ SOTO
C/VIRGEN DEL ROSARIO, 28
ALCANTARILLA (MURCIA)

En lugar de usar el típico lazo FOR Z=1 TO N: READ A: NEXT, sustitúyelo por el que lo hace en orden inverso:
FOR Z=N TO 1 STEP -1: READ A: NEXT
Es una solución sencilla a tu gran problema ¿no?

EQUIVALENCIAS



¿A qué dirección del COMMODORE-64 equivalen las siguientes, que pertenecen al VIC-20 y se entran por POKE?

POKE 30208... y POKE 30720... (esta última la podéis ver en el número 6 de la revista, en la pág. 39 y dentro del prg. "UFO" en la línea 40).

Una idea: podríais explicar a qué direcciones del C-64 equivalen las direcciones del VIC-20 ampliado (aparte de lo que apareció en la revista número 2), en este caso, por ejemplo, POKE 30000... y pico.

—Cuando comienzo a hacer un programa y pongo la instrucción:

10 PRINT" [SHIFT] [CLR]" y después listo el prg. algunas veces (últimamente bastantes) esa línea no aparece impresa.

¿A qué se puede deber esto?

Fallos frustrados del manual del usuario del C-64.

—El programa "teclado de piano" de la pág. 147 del manual de usuario del C-64 no hace nada.

1. La 30720 no es propiamente la dirección sino la que se SUMA a 7680 (la memoria de pantalla) para que dé 38400 (la memoria de color).

2. Que no aparezca una línea de programa, habiéndola introducido pulsando RETURN puede porque en vez de RETURN hayas pulsado SHIFT/RETURN, que produce el mismo efecto y no guarda la línea en la memoria, o que si estás utilizando cartuchos de ampliación (o el superexpander), que este tenga algún fallo o recalentamiento.

TECLAS DE FUNCION



Quisiera me dijesen cómo se pueden programar las teclas de funciones especiales pues en el manual del VIC-20 se hace alusión a ellos pero no pone cómo programarlos.

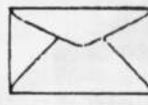
Espero que mi carta no haya sido larga pero es la primera vez que escribo a la revista.

Me despido de ustedes contando con su colaboración.

ERNESTO SARRALDE LOPEZ
C/PINTOR VERA FAJARDO, Nº 18- 3ºC
01008 VITORIA-GASTEIZ
ALAVA

Las teclas de funciones pueden ser programadas, en el número 3 se explica cómo se hace.

INSTRUCCION GOTO



Como la mayoría de los lectores estoy metido en el mundo de la informática, quizás un poquito más profundamente puesto que para mí es el sustento de mi familia. Hago referencia a la carta firmada por Guillermo Caamaño Olivares de Granada que aparece en el número 5 en la que Guillermo dice no conocer el comando "GO" al igual que la redacción.

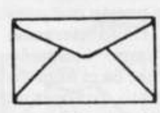
Bajo mi modesta experiencia quiero informar de lo siguiente "GO" dice una vez traducido "IR" y "TO" dice "A", tanto "GO" como "TO" son variables no utilizables puesto que las dos palabras forman la sentencia de bifurcación incondicional de un programa escrito en "BASIC" cuando decimos "GOTO-100" el programa bifurca a la línea cien, también podríamos decir "GO TO 100", traduciendo diríamos "IR A 100". Pruebe a utilizar la variable "GO" y "TO" verá cómo le dará error, también la sentencia separada tal como está escrita arriba verá como sí le funciona.

...PREGUNTONES ¿ ?

Actualmente trabajo con equipos IBM en la Empresa en la que presto mis servicios, también dispongo de un equipo CBM de propiedad motivo por el que estoy suscrito a la revista (en el MANUAL DE REFERENCIA COMMODORE hacen alusión aunque muy simple de lo referido anteriormente).

Próximamente me pondré en contacto nuevamente para así ser un colaborador más de la revista, mientras, saludos cordiales.
RICARDO ROCA
C/EXCORXADOR, 6, S/A 2º
VILAFRANCA DEL PENEDES
(BARCELONA)

JOYSTICK

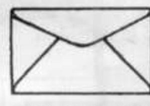


Me gustaría saber si exise alguna manera de utilizar el Joystick en un programa ya hecho (o sea un programa para utilizar el Joystick).

También quisiera intercambiar juegos con otros lectores de esta magnífica revista.
LUIS ALFONSO SANZ
Pº SAN ISIDRO, 5, 2º A.
VALLADOLID-12

Se puede utilizar. El ordenador lee la posición del joystick en las direcciones: 37137 y 37152 (VIC-20)
56320 y 56321 (C-64): Una para cada Port.

COMPATIBILIDAD DE CARTUCHOS

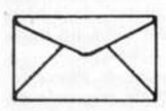


Al cargar el VICPRINT y otros programas de cinta, si además del cartucho de 16K RAM tengo puesto el super-expander, el ordenador se queda bloqueado. Si quito el super-expander todo va bien.

¿Es ésto normal?
NICOLAS DOMINGUEZ

Normalmente los dos cartuchos son compatibles, pero determinados programas que interceptan el sistema operativo KERNEL, son incompatibles con el Super-expander que también lo intercepta. Esa es la causa de tus problemas, y como ves es completamente normal en determinados programas. La solución suele ser, dejar el super-expander en el armario y sacarlo sólo cuando lo necesites de verdad.

BASE DE DATOS



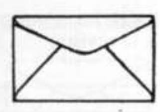
Me llamo Jorge Llambi González, tengo 13 años y poseo un Commodore 64. Paso a exponeros el problema que tenía.

En el programa base de datos de la revista nº 3 página 60 al grabar algún dato no me funcionaba, cambié la línea 400 tal como sugeríais en la revista nº 4 en meteduras de pata; pero me seguía dando error en la línea 910.

Cambié en dicha línea el primer print#j por print#1 y me dio error en la línea 920, en esta hice exactamente lo mismo: cambie el primer print#j por print#1 y funcionó.
JORDI LLAMB
APARTADO 102.
BLANES (GERONA)

Seguramente olvidaste poner el "J=1" que aparece en la línea 400 del programa. Si lo hubieses puesto habría funcionado a la perfección.

CODIGOS DE COLORES



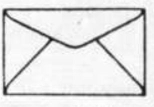
Amigos de Commodore World:
¡Hola! Me llamo Ricardo. Poseo un COMMODORE 64 desde hace unos meses, y desde entonces estoy suscrito a vuestra revista, la cual encuentro muy interesante.

Os escribo, para subsanar un fallo del manual del Usuario del COMMODORE 64. En el apéndice F, en el que se relacionan los códigos CHR$, no aparecen los correspondientes a los colores accesibles a través de la tecla LOGO de COMMODORE.

A continuación los relaciono:
CHR$ (129) Naranja.
CHR$ (149) Marrón.
CHR$ (150) Rojo claro.
CHR$ (151) Gris uno.
CHR$ (152) Gris dos.
CHR$ (153) Verde claro.
CHR$ (154) Azul claro.
CHR$ (155) Gris tres.

Y nada más por ahora. Hasta mi próxima colaboración. ¡ADIOS COMMODORIANOS!
RICARGO GARDES MOLL
C/IGLESIA, 4, 5º 13
ESPLUGAS DE LLOBREGAT (BARCELONA)

SUGERENCIAS



Hemos recibido cartas sugiriéndonos que adaptemos todos los programas publicados al otro ordenador, los del VIC-20 al C-64 y viceversa, e incluso que saquemos un par de números extra con todo lo publicado adaptado.

Como respuesta en general, los que sean colaboraciones podéis poneros en contacto con los autores, o con los clubs de ordenadores de vuestra localidad, seguramente a otro también le interese la idea y podáis compartir el trabajo entre varios, ya que nosotros no podemos emplear todo ese tiempo en adaptar programas que no son nuestros para los otros modelos de ordenadores.

En los programas que sean nuestros, casi siempre apareceran dos listados uno para el VIC-20 y otro para el C-64. Cuando no aparecen es porque el programa utiliza particularidades del sonido o gráficos del ordenador, y la adaptación sería muy difícil. También nos han sugerido que publiquemos programas para el Simons Basic en una sección exclusiva para él. Creo que esto sería injusto, pues deberíamos tener una sección FORTH, otro PASCAL, otra para el VICTREE, otra para el Berdens Basic, etc. Como no caben todas en la reista, sacaremos programas para lenguajes respectivos, pero no en todos los números.

También piden artículos sobre periféricos, hard, etc. Todo lo que nos mandan las casas distribuidoras lo sacamos en la sección Comentarios Commodore, ya iremos publicando lo que nos envíen.

Algunos piden que les enviemos listas de todo lo que hay en el mercado, como no tenemos estas listas (sólo las que vamos publicando), a continuación van las señas completas de los que desean recibir listas de productos para que las casas se las envíen: Miguel Bahi Cruz, C/Matola, Km. 2, Villacruz, nº 2. Elche (Alicante); José Luis Ranz Yubero, C/Doctor Fleming, 13, 3º izda. Guadalajara.

Además de la sugerencia del cambio entre VIC y C-64, José Luis Ranz nos pedía programas de gráficos en tres dimensiones, y quería saber cómo adaptar "evaluación de colegios" al datassette.

Los programas de gráficos están y seguirán saliendo, ya que hemos recibido una lluvia de colaboraciones sobre este tema.

El programa evaluación de colegios, no se puede adaptar para funcionar con cassette, ya que utiliza ficheros de tipo de acceso directo, y el datassete sólo puede manejar ficheros secuenciales.

(Viene de pág. 27)

continuación del listado del volcado hexadecimal

```
.:0868 56 49 43 20 42 59 20 54
.:0870 2E 20 20 48 45 4E 52 59
.:0878 22 00 93 08 06 00 9E 32
.:0880 35 36 AC C2 28 34 34 29
.:0888 AA C2 28 34 33 29 AA 31
.:0890 34 38 00 00 00 A5 2D 85
.:0898 22 A5 2E 85 23 A5 37 85
.:08A0 24 A5 38 85 25 A0 00 A5
.:08A8 22 D0 02 C6 23 C6 22 B1
.:08B0 22 D0 3C A5 22 D0 02 C6
.:08B8 23 C6 22 B1 22 F0 21 85
.:08C0 26 A5 22 D0 02 C6 23 C6
.:08C8 22 B1 22 18 65 24 AA A5
.:08D0 26 65 25 48 A5 37 D0 02
.:08D8 C6 38 C6 37 68 91 37 8A
.:08E0 48 A5 37 D0 02 C6 38 C6
.:08E8 37 68 91 37 18 90 B6 C9
.:08F0 BF D0 ED A5 37 85 33 A5
.:08F8 38 85 34 6C 37 00 BF A2
.:0900 00 00 A0 00 00 BD 93 FF
.:0908 00 F0 07 99 7E 00 00 C8
.:0910 E8 D0 F4 4C 86 E3 B0 0A
.:0918 C9 20 F0 32 38 E9 30 38
.:0920 E9 D0 8D FD FF 00 08 68
.:0928 8D FC FF 00 8E FE FF 00
.:0930 8C FF FF 00 BA BD 01 01
.:0938 C9 8C D0 07 BD 02 01 C9
.:0940 A4 F0 12 AC FF FF 00 AE
.:0948 FE FF 00 AD FC FF 00 48
.:0950 AD FD FF 00 28 60 4C 73
.:0958 00 00 AD FC FF 00 4A 90
.:0960 3A A2 00 00 86 0B A4 7A
.:0968 B9 00 00 02 38 FD 9B FF
.:0970 00 F0 13 C9 80 F0 13 E6
.:0978 0B E8 BD 9A FF 00 10 FA
.:0980 BD 9B FF 00 D0 E4 F0 C6
.:0988 E8 C8 D0 E0 84 7A A5 0B
.:0990 0A AA BD E3 FF 00 48 BD
.:0998 E2 FF 00 48 20 7B FC 00
.:09A0 4C 73 00 00 68 68 20 7B
.:09A8 FC 00 20 6B A9 4C 9F A4
.:09B0 A0 02 A5 2D D0 02 C6 2E
.:09B8 C6 2D 88 D0 F5 A5 2D 85
.:09C0 FB A5 2E 85 FC 4C 7B FD
.:09C8 00 20 29 FF 00 A9 24 8D
.:09D0 3C 03 A2 3C A0 03 A9 01
.:09D8 20 BD FF A9 0E A0 60 A2
.:09E0 08 20 BA FF 20 C0 FF A9
.:09E8 08 20 B4 FF A9 60 20 96
.:09F0 FF A9 00 00 85 90 A0 03
.:09F8 8C 3C 03 20 A5 FF 85 FD
.:0A00 A4 90 D0 39 20 A5 FF 85
.:0A08 FE A4 90 D0 30 AC 3C 03
.:0A10 88 D0 E5 A6 FD A5 FE 20
.:0A18 CD BD 20 78 FF 00 20 A5
.:0A20 FF A6 90 D0 19 C9 00 00
.:0A28 F0 0E 20 D2 FF 20 E1 FF
.:0A30 F0 0D 20 66 FF 00 4C 42
.:0A38 FD 00 20 75 FF 00 A0 02
.:0A40 D0 BB A9 0E 20 C3 FF 20
.:0A48 CC FF 4C 86 E3 20 2C FF
.:0A50 00 4C 86 E3 A5 2B 85 FB
.:0A58 A5 2C 85 FC 20 ED FE 00
.:0A60 20 D8 FE 00 A9 00 00 A6
.:0A68 FB A4 FC 20 D5 FF 86 2D
.:0A70 B4 2E EA EA EA EA EA EA
.:0A78 EA EA EA EA 20 33 A5 20
.:0A80 59 A6 4C B8 FD 00 20 ED
.:0A88 FE 00 20 D8 FE 00 A5 2B
.:0A90 85 FB A5 2C 85 FC A9 FB
.:0A98 A6 2D A4 2E 20 D8 FF 20
.:0AA0 24 FF 00 20 92 FE 00 4C
.:0AA8 86 E3 20 39 FF 00 20 EF
.:0AB0 FE 00 20 73 00 00 C9 2C
.:0AB8 D0 2D 9D 3C 03 E8 20 73
.:0AC0 00 00 C9 49 D0 22 20 73
.:0AC8 00 00 F0 1D 9D 3C 03 E8
.:0AD0 20 73 00 00 F0 14 9D 3C
.:0AD8 03 E8 20 73 00 00 D0 0B
.:0AE0 9D 3C 03 20 4E FF 00 D0
.:0AE8 09 4C 71 FE 00 4C 08 AF
.:0AF0 20 29 FF 00 4C 86 E3 20
.:0AF8 E2 FC A2 04 4C 41 FC 00
.:0B00 20 3C FF 00 20 79 00 00
.:0B08 F0 E7 C9 22 D0 E3 A5 7A
.:0B10 85 FB A5 7B 85 FC 20 73
.:0B18 00 00 F0 D6 C9 22 D0 F7
.:0B20 20 73 00 00 C9 54 D0 CB
.:0B28 20 73 00 00 C9 4F D0 C4
.:0B30 20 73 00 00 20 EF FE 00
.:0B38 A9 3D 9D 3C 03 E8 A5 FB
.:0B40 85 7A A5 FC 85 7B 20 EF
.:0B48 FE 00 A9 00 00 9D 3C 03
.:0B50 4C 71 FE 00 20 3F FF 00
.:0B58 20 EF FE 00 A9 00 00 9D
.:0B60 3C 03 20 4E FF 00 D0 29
.:0B68 4C 71 FE 00 20 ED FE 00
.:0B70 A9 00 00 9D 3C 03 20 29
.:0B78 FF 00 20 1A FF 00 A0 00
.:0B80 00 B9 3C 03 F0 06 20 A8
.:0B88 FF C8 90 F5 20 AE FF 20
.:0B90 92 FE 00 F0 03 20 92 FE
.:0B98 00 4C 86 E3 A9 08 85 BA
.:0BA0 20 B4 FF A9 6F 20 96 FF
.:0BA8 A0 00 00 20 A5 FF 99 3C
.:0BB0 03 C8 C9 0D D0 F5 A9 00
.:0BB8 00 99 3C 03 20 AB FF A0
.:0BC0 00 00 A9 30 D9 3C 03 D0
.:0BC8 06 C8 D9 3C 03 F0 15 20
.:0BD0 24 FF 00 20 CC FF 20 75
.:0BD8 FF 00 A9 3C A0 03 20 1E
.:0BE0 AB 68 68 4C 86 E3 60 20
.:0BE8 29 FF 00 8A A2 3C A0 03
.:0BF0 20 BD FF A9 08 AA A0 00
.:0BF8 00 84 90 4C BA FF A2 00
.:0C00 00 86 B7 20 79 00 00 F0
.:0C08 1F C9 22 D0 1B E6 7A D0
.:0C10 02 E6 7B A0 00 00 B1 7A
.:0C18 F0 0A C9 22 F0 06 9D 3C
.:0C20 03 E8 D0 EA E4 B7 F0 01
.:0C28 60 68 68 4C 08 AF A9 08
.:0C30 20 B1 FF A9 6F 4C 93 FF
.:0C38 A9 08 4C C3 FF A9 49 2C
```



```
.:0C40 A9 56 48 20 1A FF 00 68
.:0C48 20 A8 FF 4C AE FF A9 4E
.:0C50 2C A9 52 2C A9 53 A2 00
.:0C58 00 9D 3C 03 E8 A9 3A 9D
.:0C60 3C 03 E8 60 A2 00 00 BD
.:0C68 7D FF 00 F0 06 20 16 E7
.:0C70 E8 D0 F5 20 CF FF C9 53
.:0C78 08 20 75 FF 00 28 60 20
.:0C80 E4 FF F0 09 C9 20 D0 05
.:0C88 20 E4 FF F0 FB 60 A9 0D
.:0C90 2C A9 20 4C D2 FF 0D 45
.:0C98 53 54 41 53 20 53 45 47
.:0CA0 55 52 4F 3F 20 28 53 2F
.:0CA8 4E 29 20 00 00 4C 52 FC
.:0CB0 00 00 00 B0 0A C9 00 00
.:0CB8 41 50 50 45 4E C4 43 41
.:0CC0 54 41 4C 4F C7 43 4F 4C
.:0CC8 4C 45 43 D4 44 4C 4F 41
.:0CD0 C4 44 53 41 56 C5 48 45
.:0CD8 41 44 45 D2 49 4E 49 D4
.:0CE0 4B 49 4C CC 4F 46 C6 52
.:0CE8 45 4E 41 4D C5 53 43 52
.:0CF0 41 54 43 C8 53 45 4E C4
.:0CF8 53 54 41 54 55 D3 00 00
.:0D00 D7 FC 00 EF FC 00 6C FD
.:0D08 00 72 FD 00 A0 FD 00 C0
.:0D10 FD 00 FD FD 00 03 FE 00
.:0D18 06 FE 00 0B FE 00 55 FE
.:0D20 00 68 FE 00 8B FE 00 00
.:0D28 00 00 00 00 00 00 00 00
.
```


Las Aventuras de RAMI y ROMO
Kbit y Kbita
La Batalla del Abuelo
por ERRAZQUIN
LE HE PEDIDO AL DIBUJANTE QUE ME DEJE EN ESTA HISTORIA CONTAROS ALGUNAS COSAS SOBRE AQUELLOS SABIOS, QUE CON SU TRABAJO, HAN CONTRIBUIDO A QUE HOY DIA PODAMOS CONTAR CON LOS ORDENADORES
EL HOMBRE SIEMPRE TUVO LA NECESIDAD DE CONTAR... Y EMPEZO CON LO MAS SENCILLO Y PROXIMO QUE TENIA... ¡LOS DEDOS!


EN LA ANTIGUA MESOPOTAMIA SE UTILIZABAN ABACOS ... ERA UN SISTEMA DE CALCULO POR BOLAS QUE INDICABAN LAS DECENAS, ETC
IMPUESTOS
NUEVO SISTEMA
ABACO
100
20
9


PARECIDO SISTEMA AL METODO EMPLEADO EN LA ANTIGUA CHINA ... ¿CURIOSO, NO?


DE LOS PUEBLOS ANTIGUOS, LOS DE MAYOR DESARROLLO EN EL PLANO MATEMATICO FUERON LOS EGIPCIOS Y LOS MAYAS
MI CALENDARIO ES DE 365 DIAS EXACTOS
PUES EL MIO ES DE 365,2420
¡OH!


LOS GRIEGOS Y LOS ROMANOS NO TENIAN NUMEROS PERO USABAN LETRAS PARA CONTAR ....
COPIONES
EL IMPERIO ES EL IMPERIO
M = 1000
D = 500
C = 100
L = 50
X = 10 e I = 1


LOS NUMEROS ACTUALES SON UN INVENTO ARABE
OS HEMOS GANADO 103 BATALLAS, CONTAD SI PODEIS.. JI, JI

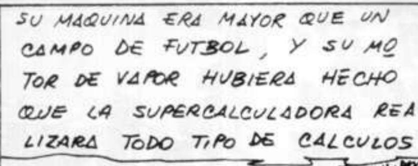

GRACIAS AL SISTEMA NUMERICO ARABE LA CULTURA MEDIEVAL HACE QUE LA MATEMATICA, LA FISICA, LA ASTRONOMIA Y LA QUIMICA SE DESARROLLEN..
3 x 6 = 119 ¿NO?... GRRMBBLL ¡MIL RAYOS!


HABRA QUE ESPERAR AL AÑO 1642 PARA QUE BLAISE PASCAL CONSTRUYA UNA MAQUINA DE CALCULAR CON RUEDECILLAS DENTADAS
MON DIEU
RACC RICC RACC
VES PAPI EN UN MOMENTO TU DECLARACION DE LA RENTA


GOTTFRIED LEIBNIZ EN 1673 PERFECCIONA LA MAQUINA DE PASCAL Y ESTUDIA LA MATEMATICA BINARIA DESDE UN PLANO FILOSOFICO
TODO PUEDE SER TRATADO EN BASE A CEROS Y UNOS.. ALGO EXISTE.. ES UN UNO, NO EXISTE ...UN CERO
1: EXISTENCIA
0: NO EXISTENCIA


JACQUARD EN 1801 CONSTRUYE UN TELAR QUE EFECTUA DIBUJOS EN LAS TELAS GRACIAS A UNA TARJETA QUE LLEVABA UNOS AGUJEROS EN SITIOS ESPECIALES
VOILA C'EST MAGNIFIQUE





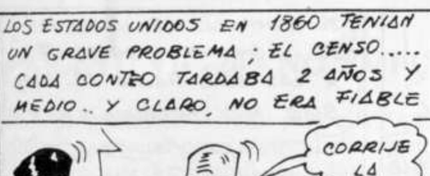






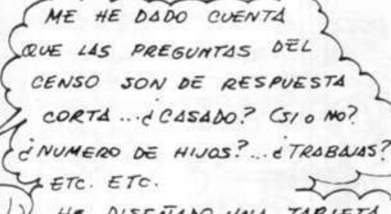








CON 63.000.000 DE PERSONAS Y LA CONTINUA INMIGRACION ERA IMPOSIBLE CONOCERLO CON PRECISION, PERO APARECIO EN ESCENA HOLLERITZ

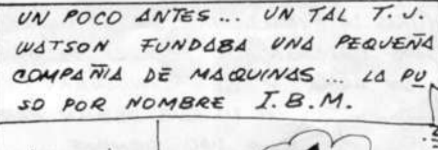















LOS ORDENADORES ESTABAN A PUNTO DE SER CONSTRUIDOS... PERO UN ACONTECIMIENTO HISTORICO HIZO QUE SU ENTRADA FUESE AUN MAS RAPIDA


LA IIª GUERRA MUNDIAL
1939
1945


... Y CADA PAIS INTENTÓ DESARROLLAR UN ORDENADOR QUE AYUDASE A GANARLA
COSAS DE LA GUERRA
EN EL REINO UNIDO ALAN M. TURING ...PARA LA INTERPRETACION DE MENSAJES CIFRADOS
BIIP BIP
EN ALEMANIA LAS MAQUINAS TIPO "Z" DEL DOCTOR ZUSE
... Y EN ESTADOS UNIDOS EL EQUIPO IBM, EL DOCTOR AIKEN Y LA UNIVERSIDAD DE PENN-SILVANIA


EL PRIMER ORDENADOR SE LLAMÓ ASCC o MARK-I, CONSTRUIDO POR AIKEN, MEDIA 59 PIES DE LARGO POR 8 DE ALTO. SE COMPONÍA DE MAS DE 800.000 PIEZAS
LO HICE LO HICEEEE AUAAAAHH


MAS WATSON Y LA UNIVERSIDAD DE PENNSILVANIA CONSTRUIAN POCO DESPUES EL ENIAC (1946). QUE ERA TAMBIEN DE GRANDES DIMENSIONES Y DE UNAS CARACTERISTICAS SIMILARES
ESTO MARCA EL COMIENZO DE UNA FRUCTIFERA UNION ... ¡OJALA!
IBM


SE COMPONIA DE VALVULAS DE VACIO INCANDESCENTES A LAS QUE HABIA DE CAMBIAR MUY FRECUENTEMENTE
PERO EL PASO ESTABA DADO LOS ORDENADORES FUNCIONABAN
¡AH! .. POR CIERTO.. YO SOY UNA VALVULA DE VACIO... JI, JI, JI


TRAS LA FINALIZACION DE LA GUERRA LOS ORDENADORES DEJARON DE SER UN ARMA Y SE CONVIRTIERON EN LO QUE DEBIAN HABER SIDO SIEMPRE ... ¡UNA HERRAMIENTA!
OTRO DIA OS HABLARÉ DE LOS ORDENADORES MODERNOS... PERO HASTA ENTONCES...


ME GUSTARIA QUE RECORDARAIS A ESTOS SABIOS Y A OTROS TANTOS QUE NO HAN PASADO A LA HISTORIA... SIN ELLOS NO TENDRIAMOS HOY DIA INFORMATICA... NI OTRAS MUCHAS COSAS
CARAZQUIN

VIDEO CASINO

Carrera de caballos

Gabe GARGIULO (RUN EEUU)

Traducido por Valerie SHANKS y adaptado por Alvaro IBAÑEZ

Guarda tus prismáticos y vente al hipódromo (con este programa y tu VIC-20 lo podrás hacer sin perder dinero).

PARA TU VIC-20 - NO AMPLIADO

Este programa de carreras para el VIC-20 no ampliado utiliza el símbolo que se parece un poco a un caballo, o un perro o un pollo, con un poco (a decir verdad, con cantidades industriales) de imaginación. Mi versión sólo funciona en el VIC, ya que utiliza pokes.

Tú empiezas a jugar con $500, y puedes apostar hasta esa cantidad. Tú escoges un caballo, de 1 a 5, para ganar. El programa elige el ganador al azar. Si tú has apostado por el ganador, ganas cuatro veces la cantidad apostada, que se añade a la cantidad que tienes. Si pierde tu caballo, tú pierdes la cantidad apostada. Puedes jugar hasta que pierdas todo el dinero que tenías. (Esto es inevitable.)

Es fácil teclear este listado, el juego resulta divertido, y sobre todo, no te cuesta nada. El listado demuestra un buen estilo de programación, que si lo sigues, da buenos resultados que son fáciles de comprender y modificar. Empieza con un REM (comentario) que demuestra el nombre y finalidad del programa. Ahora lista las variables y su explicación. A continuación, presenta la lógica principal del programa, que contiene sus decisiones más importantes. Coloca las subrutinas, que se utilizan con las sentencias GOSUB, después de la lógica principal del programa.



Tabla 1. Descripción del programa principal

Nº DE LINEA	
15	FIJA R1 (CANTIDAD DE DINERO EN MANO) EN 500
20	BORRA LA PANTALLA. LLAMA LA SUBRUTINA 1200 PARA COLOCAR UNA FILA DE GUIONES ATRAVESANDO LA PANTALLA. DESCRIBE LAS VARIABLES ML, LA.
22	FIJA ML (POSICION DE MEMORIA) EN 7680, LA DIRECCION INICIAL DE LA PARTE SUPERIOR IZQUIERDA DE LA PANTALLA. FIJA LA (líneas transversales) EN 22. POKE 36879: FIJA EL COLOR Y EL FONDO.
40	LLAMA LA SUBRUTINA 1200 PARA QUE COLOQUE UNA LINEA DE GUIONES EN LA PANTALLA.
80	PONE EL TITULO Y LLAMA LA SUBRUTINA 1200 PARA QUE COLOQUE UNA FILA DE GUIONES EN LA PANTALLA.
85	DESPUES DE UNA CORTA ESPERA, ARRANCA EL PROGRAMA.
86	FIJA LA POSICION INICIAL DE LOS CABALLOS.
90 - 110	DA LAS INSTRUCCIONES.
140	LLAMA LA SUBRUTINA 1200 PARA QUE COLOQUE UNA FILA DE GUIONES EN LA PANTALLA.
270	APUNTA EL NUMERO DEL CABALLO APOSTADO.
300	LLAMA LA SUBRUTINA 1200 PARA QUE COLOQUE UNA FILA DE GUIONES EN LA PANTALLA.
310	SOLICITA LA APUESTA.
320	SI LA APUESTA ES MENOR QUE O IGUAL A LA CANTIDAD EN MANO, SE AVANZA A LA 400.
330	(Y SI NO LO ES). INDICA LA CANTIDAD QUE QUEDA PARA APOSTAR.

350 SOLICITA UNA APUESTA DE NUEVO. (310.)
400 BORRA LA PANTALLA Y LLAMA LA SUBRUTINA 2000 PARA QUE PONGA LOS CABALLOS.
410 LLAMA LA SUBRUTINA 4000 PARA QUE PONGA LA PUERTA DE SALIDA.
600 ELIGE UN NUMERO AL AZAR ENTRE 1 Y 5.
620 LLAMA LA SUBRUTINA 1000 PARA QUE 1 SE SUME A UN CONTADOR QUE CORRESPONDE AL CABALLO GANADOR. LLAMA LA SUBRUTINA 2000 PARA QUE PONGA LOS CABALLOS EN LA PANTALLA. (SI EL CONTADOR DE UN CABALLO HA SIDO INCREMENTADO, ESTE AVANZA.)
630 SE SUMA 1 AL CONTADOR QUE CORRESPONDE AL CABALLO GANADOR.
640 SI EL CABALLO QUE ACABA DE DESPLAZARSE NO SE ENCUENTRA CERCA DEL LADO DERECHO DE LA PANTALLA, EL PROGRAMA VA A LA 600 PARA QUE SE MUEVA OTRO CABALLO.
650 SI HA HABIDO GANADOR, EL PROGRAMA ESPERA UN POCO AQUI.
660 ANUNCIA AL GANADOR.
665 COLOCA UNA FILA DE GUIONES POR LA PANTALLA.
670 SI EL CABALLO ELEGIDO ES EL GANADOR, SE SUMAN LAS GANANCIAS A LA CANTIDAD EN MANO; EL PROGRAMA AVANZA A LA 750.
680 SI EL CABALLO ELEGIDO NO ES EL GANADOR, EL PROGRAMA HACE UNA PAUSA AQUI. APARECE EN PANTALLA EL MENSAJE "TU PIERDES". RESTA LA CANTIDAD DE LA APUESTA DE LA CANTIDAD DE DINERO EN MANO.
685 INDICA LA CANTIDAD DE DINERO QUE QUEDA.
687 SI NO QUEDA DINERO, APARECE EL MENSAJE "ESTAS BROKE". EL PROGRAMA TERMINA.
690 TE PREGUNTA SI QUIERES JUGAR OTRA VEZ.
691 RECIBE RESPUESTA.
700 SI LA RESPUESTA ES "S", EL PROGRAMA VUELVE A LA 20 PARA EMPEZAR DE NUEVO.
710 SI LA RESPUESTA NO ES "S", ACABA AQUI. RESTAURA EL COLOR DE PANTALLA Y EL FONDO. FINALIZA EL PROGRAMA.
750 PONE "TU GANAS" Y TE DICE LA CANTIDAD.
760 PONE LA CANTIDAD DE DINERO EN MANO.
770 EL PROGRAMA VUELVE A LA 690 PARA PREGUNTAR SI QUIERES JUGAR DE NUEVO.
990 FINALIZA.

Tabla 2. Descripciones de las subrutinas

SUBRUTINAS
1000 SUMA 1 A X1, X2, X3, X4 O X5, SEGUN EL NUMERO QUE HAYA SALIDO AL AZAR.
1200 COLOCA UNA FILA DE ASTERISCOS POR LA PANTALLA.
2000 HACE AVANZAR LOS CABALLOS CUYO NUMERO HA SALIDO. DEJA LOS OTROS CABALLOS EN SU SITIO.
3000 PRODUCE EL RUIDO DE LA PISTOLA DE SALIDA Y EL GALOPE DE LOS CABALLOS.

Tabla 3. Definiciones de las variables

VARIABLES	
M1	POSICION EN LA MEMORIA DE UN CABALLO.
LA	LINEAS HORIZONTALES EN PANTALLA, 22 EN EL VIC
X1	POSICION DEL CABALLO 1
X2	POSICION DEL CABALLO 2
X3	POSICION DEL CABALLO 3
X4	POSICION DEL CABALLO 4
X5	POSICION DEL CABALLO 5
R1	CANTIDAD DE DINERO EN MANO.
I	VARIABLE DE INDICE
R	NUMERO ELEGIDO AL AZAR.
H	CABALLO POR EL QUE SE HA APOSTADO.
B	CANTIDAD APOSTADA.
B( )	"ARRAY" EMPLEADO PARA PODER LOCALIZAR A CADA CABALLO. B(1) ES PARA EL CABALLO #1, B(2) ES PARA EL CABALLO #2, ETC.
Z$	RESPUESTA S/N
J	VARIABLE DE INDICE
Y	COORDENADA "Y" EN PANTALLA. (EL NUMERO DE LINEAS A PARTIR DE LA PARTE SUPERIOR).
J3	VARIABLE DE INDICE.
Y	VARIABLE DE INDICE.
J2	VARIABLE DE INDICE.
L	VARIABLE DE INDICE.
M	VARIABLE DE INDICE.

VIC-20

```
10 REM CARRERAS
15 R1=500:REM R1=DINERO QUE QUEDA
20 PRINT"[CLR][WHT]":REM ML=LOCALIZA
CION DE LA PANTALLA
22 ML=7680:LA=22:POKE36879,8
30 B=0:R=0:FORI=1TO5:A(I)=0:B(I)=0:N
EXTI
40 GOSUB1200
80 PRINT"BIENVENIDO[SPC]A[SPC]LAS[9SPC]
CARRERAS[2SPC]":GOSUB1200
85 FORI=1TO500:NEXT:REM AQUI COMIENZ
A EL JUEGO
86 X1=1:X2=1:X3=1:X4=1:X5=1:REM COLO
CA A LOS CABALLOS
90 PRINT"TIENES[SPC]$";R1;"PARA":PRI
NT"APOSTAR."
100 PRINT"LAS[SPC]APUESTAS[SPC]ESTAN
[5SPC]4:1":PRINT
110 PRINT"ELIGE[SPC]UN[SPC]CABALLO[SPC]
(1-5)";
140 GOSUB1200
270 INPUT"[SPC]";H
280 IFH>0ANDH<6THEN310
290 PRINT"LOS[SPC]CABALLOS[SPC]ESTAN
[4SPC]NUMERADOS[SPC]DEL[SPC]1[SPC]AL
[SPC]5.[SPC]INTRODUCE[SPC]EL[SPC]NUM
ERO"
300 GOSUB1200:GOTO270
310 PRINT"CUANTO[SPC]APUESTAS";:INPU
TB:B=ABS(B)
320 IFB<R1 OR B=R1THEN400
330 PRINT"TIENES[SPC]$";R1;".[SPC]";

350 GOTO310
400 PRINT"[CLR]";:GOSUB2000:REM COLO
CA LOS CABALLOS EN SU SITIO
410 GOSUB4000:REM LINEA DE SALIDA
600 R=INT(5*RND(1)+1)
620 GOSUB1000:GOSUB2000
630 B(R)=B(R)+1
640 IFB(R)<(LA-2)THEN600
650 PRINT"[SPC]":FORI=1TO1000:NEXTI
660 PRINT"[CLR][2CRSRD]EL[SPC]GANADO
R[SPC]ES[SPC]EL[6SPC]NUMERO";R
665 FORI2=1TOLA:PRINT"-";:NEXTI2
670 IFH=RTHENR1=R1+(4*B):GOTO750
680 PRINT"HAS[SPC]PERDIDO.":R1=R1-B
685 PRINT"[CRSRD]TIENES[SPC]$";R1
687'IFR1<=0THENPRINT"[CRSRD]ESTAS[SPC]
ARRUINADO.VUELVEOTRO[SPC]DIA":END
690 Z$="":PRINT"[CRSRD]OTRA[SPC]CARR
ERA(S/N)";
691 INPUTZ$
700 IFZ$="S"THEN20
710 POKE36879,27:PRINT"[CLR][BLU]";:
END
750 PRINT"[CRSRD]HAS[SPC]GANADO[SPC]
$";4*B
760 PRINT"[CRSRD]AHORA[SPC]TIENES[SPC]
$";R1
770 GOTO690
990 END
1000 REM SUMA AL CONTADOR APROPIADO
1090 IFR=1THENX1=X1+1
1092 IFR=2THENX2=X2+1
1093 IFR=3THENX3=X3+1
1094 IFR=4THENX4=X4+1
1095 IFR=5THENX5=X5+1
1100 RETURN
1200 PRINT:FORJ=1TOLA:PRINT"-";:NEXT
J:PRINT:RETURN
2000 REM COLOCA LOS CABALLOS EN LA P
ANTALLA
2010 Y=1:PRINT"[CLR]"
2020 POKEML+X1+22*Y,94
2021 POKEML+X1-1+22*Y,48+1
2022 POKEML+X1-2+22*Y,32
2030 Y=5
2040 POKEML+X2+22*Y,94
2041 POKEML+X2-1+22*Y,48+2
2042 POKEML+X2-2+22*Y,32
2050 Y=9
```

```
2060 POKEML+X3+22*Y,94
2061 POKEML+X3-1+22*Y,48+3
2062 POKEML+X3-2+22*Y,32
2070 Y=13
2080 POKEML+X4+22*Y,94
2081 POKEML+X4-1+22*Y,48+4
2082 POKEML+X4-2+22*Y,32
2090 Y=17
2100 POKEML+X5+22*Y,94
2101 POKEML+X5-1+22*Y,48+5
2102 POKEML+X5-2+22*Y,32
2130 RETURN
```

```
3000 REM SONIDO
3010 POKE36878,15:POKE36874,200
3020 FORJ3=1TO5:NEXTJ3:POKE36874,0
3030 POKE36878,0:RETURN
4000 REM SONIDO PARA COMENZAR EL JUE
GO Y DISPARO
4010 FORY=0TO22:POKEML+3+22*Y,115:NE XTY
4020 FORJ2=1TO1011:NEXTJ2
4040 POKE36877,220:FORL=10TO0STEP-1
4050 POKE36878,L:FORM=1TO70:NEXTM:NE
XTL:POKE36877,0:POKE36878,0
4100 RETURN
```

ESTE ES PARA TU C-64

A prueba de reflejos

Soltan SZEPESI (RUN EEUU)

Traducido por Valerie SHANKS y adaptado por Diego ROMERO

Unas secuencias de color y sonido ponen a prueba tu memoria y velocidad de reflejos en este divertido programa.



El programa "Repetir la Secuencia" te permite jugar a tres cosas distintas que te hacen ejercer y te ayudan a mejorar tu memoria tanto visual como auditiva. Otro juego de este programa resulta útil para comprobar tus reflejos. El programa (Listado 1) está escrito en Basic para el C-64. pero puede ser modificado para ser usado en otros equipos.

La idea detrás de estos juegos es la misma que la de "Simón". Sin embargo, al utilizar el ordenador y su pantalla, se logra una mayor comunicación entre la máquina y el jugador.

Cuatro cuadrados de diferentes colores aparecen en cuatro sitios distintos en pantalla. Salen unos colores intermitentes aleatorios, casa uno acompañado por un sonido especial. Tú lo tienes que repetir pulsando la misma tecla de color (sin usar la tecla de control). Los colores utilizados son púrpura, verde, azul y amarillo: por lo tanto, las teclas 5, 6, 7 y 8 tienen que ser pulsadas. Si repites los colores y los sonidos correctamente, el juego continúa. A continuación se presenta la descripción de los cuatro juegos.

Juego 1. Crear la secuencia.

Después de haber recibido la primera señal del C-64, tienes que añadir otra señal. A continuación, tienes que repetir la secuencia de las señales anteriores y añadir otra. Continúas de esta forma hasta completar un número determinado de pasos (ver la Tabla 1), y el C-64 te saludará con las primeras ocho notas de la Quinta Sinfonía de Beethoven. Si no lo has hecho con suficiente rapidez, o si te has equivocado al repetir la secuencia, el ordenador te toca una parte ruidosa de la Quinta Sinfonía y el juego termina.

Antes de empezar cada juego, puedes elegir uno de los cuatro niveles de dificultad pulsando una de las teclas de funciones programables (F1, F3, F5 o F7). La Tabla 1 lista los distintos parámetros definidos por estas teclas.

Cuando se pulsa la tecla de función F3 (después de que el C-64 lo solicite al iniciarse el juego), se tienen que repetir ▶

correctamente ocho secuencias para poder terminar el juego. El tiempo transcurrido entre cada señal será muy corto; tienes que pulsar el color correcto a los tres segundos de haber pulsado el color anterior.

Juego 2. Repetir la secuencia.

El C-64 empieza por dar una señal. Al repetir tú la señal correctamente, el ordenador repite la señal anterior y añade una señal nueva. Tienes que volver a repetir esta secuencia de nuevo. En los pasos siguientes, el 64 repite la secuencia anterior y añade una señal nueva hasta que se complete la serie de acuerdo con el número de secuencias elegido.

Juego 3. Cambiar la secuenca.

Este juego se parece mucho al Juego 2. La única diferencia es que el ordenador no repite la secuencia dada anteriormente, sino que empieza una nueva secuencia con una señal adicional.

Juego 4. Intermitencias Simples

En este juego, el C-64 sólo da una señal a la vez y tú dispones de un segundo para repetirla.
La Tabla 2 presenta los números de las secuencias y el contenido de cada sección del programa. La lista de las variables principales se presenta en la Tabla 3.

Las variables N1, T1 y T2 son fijadas por las cuatro teclas programables en las sentencias 60 a 75. Pueden ser modificadas si se cambian los números en estas sentencias. El tiempo transcurrido entre cada señal (T1) es totalmente un valor relativo. Este se añade al tiempo que se tarda en representar el cuadrado de color. T2 está en segundos. En el Juego 4, T2 se vuelve a definir en la sentencia 355.
Dado que el programa sale de la subrutina "REPITE LA SECUENCIA" (las secuencias 450 a 485) en unas circunstancias determinadas sin utilizar el comando "Return", el "stack" puede llenarse (después de unas 13 partidas en el mismo número de juego) y podría presentarse un mensaje de error "OUT OF DATA". Para evitar este desastre, se introdujeron tres comandos adicionales "Return" con la bandera Q. La subrutina "FOR STACK CLEARING" (las sentencias 700 a 710) proporciona la sentencia correcta "GOTO" (de salto).

Una operación semejante de llenar los "stacks" puede realizarse cuando queda un bucle For... Next antes de terminarlo. La primera parte de la sentencia 515 soluciona este problema.
El programa requiere 3568 bytes de memoria; se necesitan 400 bytes adicionales para ejecutar el programa.

Tabla 1. Los niveles de dificultad dentro de un juego determinado

TECLA DE FUNCIONES	NUMERO DE SECUENCIAS	TIEMPO ENTRE SEÑALES	TIEMPO PERMITIDO AL JUGADOR
F1	4	100	5 SEGUNDOS
F3	8	50	3 SEGUNDOS
F5	16	10	2 SEGUNDOS
F7	32	1	1 SEGUNDO

Tabla 2. Lista de secciones principales del programa

SENTENCIA Nº	TEMA Y COMENTARIOS
5-10	TITULO Y AUTOR
15-95	INICIALIZACION. ELEGIR NUMERO DE JUEGO Y NIVEL DE DIFICULTAD.
200-230	PROGRAMA PRINCIPAL DEL JUEGO 1
250-275	PROGRAMA PRINCIPAL DEL JUEGO 2
300-310	PROGRAMA PRINCIPAL DEL JUEGO 3
350-360	PROGRAMA PRINCIPAL DEL JUEGO 4
400-435	SUBRUTINA DE COLOR Y SONIDO
450-485	SUBRUTINA PARA REPETIR LA SECUENCIA
500-520	MENSAJES DE ERROR
525-560	PROGRAMA DE MUSICA
570-580	FINAL CORRECTO. QUINTA SINFONIA. (DATOS DE LAS LINEAS 555-560)
600-640	REPETICION DE LA ULTIMA SECUENCIA CORRECTA?
650-680	QUIERES CONTINUAR?
700-710	BORRADO DE LOS "STACKS"

Tabla 3. Lista de las variables principales

VARIABLE	COMENTARIOS
B$	11 CURSOR ABAJO + CODIGO PURPURA
C$	MARCA LAS MANCHAS DE COLOR
CO	POSICION DE MEMORIA DE COLOR
G	NUMERO DE PARTIDAS DENTRO DEL MISMO TIPO DE JUEGO.
K(N)	POSICION DE LA MANCHA DE COLOR EN LA INTERMITENCIA N DE LA SECUENCIA.
N1	MAXIMO NUMERO DE INTERMITENCIAS EN UNA SECUENCIA.
NM	NUMERO DE NOTAS EN UNA SECUENCIA DETERMINADA
P	NUMERO DE INTERMITENCIAS DEL JUEGO 4
Q	BANDERA AL SALIR DE UNA SUBRUTINA
S	POSICION DE MEMORIA DE PANTALLA
SI	DURACION DE LA MUSICA
T	NUMERO DE DIRECCION DEL SONIDO
T1	TIEMPO TRANSCURRIDO ENTRE SEÑALES (VER EXPLICACION)
T2	RETRASO DE TIEMPO PERMITIDO, EN SEGUNDOS, AL REPETIR LA SEÑAL
T3	ESTADO DEL RELOJ, AL MEDIR RETRASO DE TIEMPO T2
TN	CODIGO DEL TONO DE LAS NOTAS MUSICALES
TT	NUMERO TOTAL DE PUNTOS DENTRO DEL MISMO TIPO DE JUEGO
X	NUMERO DE JUEGO

C-64

```
5 REM
10 REM
14 REM**INICIALIZACION***************
***
15 POKE53281,0:DIMK(32):G=0:TT=0:PRI
NT"[WHT]"
20 PRINT"[CLR][2CRSRD]**[SPC]MODALID
ADES[SPC]DE[SPC]JUEGO:[15SPC][CRSRD]
[5SPC]1.SECUENCIA[SPC]PROPIA"
25 PRINT"[CRSRD][4SPC]2.REPETICION[SPC]
DE[SPC]SECUENCIA"
30 PRINT"[CRSRD][4SPC]3.SECUENCIA[SPC]
CAMBIANTE[14SPC][CRSRD][5SPC]4.DESTE
LLOS[SPC]SUELTOS"
35 PRINT"[2CRSRD]**[SPC]PULSA[SPC]EL
[SPC]NUMERO:";
40 GETA$:IFA$=""THEN40
45 X=VAL(A$):PRINTX:IFX<1ORX>4THENRU
N
50 PRINT"[2CRSRD]**[SPC]NIVEL[SPC]DE
[SPC]DIFICULTAD[SPC](CRECIENTE)[6SPC]
[CRSRD][2SPC]PULSA[SPC]F1,[SPC]F3,F5
[SPC]O[SPC]F7"
55 GETA$:IFA$=""THEN55
60 IFA$=CHR$(133)THENT1=100:N1=4:T2=
5
65 IFA$=CHR$(134)THENT1=50:N1=8:T2=3
```

```
70 IFA$=CHR$(135)THENT1=10:N1=16:T2=
2
75 IFA$=CHR$(136)THENT1=1:N1=32:T2=1
77 IFT2=0THENRUN
80 B$="[10CRSRD][PUR]":C$="[2SPC]PUR
PURA[3SPC][GRN]VERDE[3SPC][BLU]AZUL[3SPC]
[YEL]AMARILLO"
85 V=54296:WA=54276:A=54277:SR=54278
:HF=54273:LF=54272
90 PH=54275:PL=54274:S=1227:CO=55499
95 NM=1:G=G+1:Q=0
100 PRINT"[CLR]"B$C$:W=RND(-TI)
105 ONXGOTO200,250,300,350
199 REM**SECUENCIAS PROPIAS*********
***
200 PRINT"[CRSRD]EL[SPC]C-64[SPC]DA[SPC]
EL[SPC]PRIMER[SPC]COLOR" :N=1:K(1)=I
NT(RND(1)*4):GOSUB400
205 GOSUB450:IFQ<>0THEN700
210 NM=NM+1:N=NM:PRINT"[CRSRD]AGREGA
[SPC]OTRO[SPC]COLOR":T3=TI
215 GETA$:IFA$=""ANDTI-T3<T2*60THEN2
15
220 IFTI-T3>T2*50THEN500
225 K(N)=VAL(A$)-5:GOSUB400:GOSUB450
:IFQ<>0THEN700
230 GOTO210
249 REM**REPETICION DE SECUENCIAS***
***
250 PRINT"[CRSRD]EL[SPC]C-64[SPC]DA[SPC]
LA[SPC]SECUENCIA":N=NM:K(N)=INT(RND(
1)*4):GOSUB400
255 GOSUB450:IFQ<>0THEN700
260 FORI=1TO800:NEXT
265 FORN=1TONM:PRINT"[CRSRD]EL[SPC]C
-64[SPC]DA[SPC]LA[SPC]SECUENCIA":GOS
UB400:NEXTN
270 NM=NM+1:GOTO250
275 FORI=1TO800:NEXT
299 REM**SECUENCIAS CAMBIANTES******
***
300 FORN=1TONM:PRINT"[CRSRD]EL[SPC]C
-64[SPC]DA[SPC]LA[SPC]SECUENCIA":K(N
)=INT(RND(1)*4)
305 GOSUB400:NEXTN:GOSUB450:IFQ<>0TH
EN700
310 FORI=1TO800:NEXTI:NM=NM+1:GOTO30
0
349 REM**DESTELLOS SUELTOS**********
***
350 P=1
355 T2=1:N=1:K(N)=INT(RND(1)*4):GOSU
B400:GOSUB450:IFQ<>0THEN700
360 FORI=1TO400:NEXTI:NM=1:N=1:P=P+1
:GOTO355
399 REM**DESTELLOS COLOR AND SOUND**
****
400 POKEV,15:POKEA,0:POKESR,240:POKE
HF,20+K(N)*5:POKELF,10:POKEWA,17
405 PRINT"[HOM]"B$+"[4CRSRD]PASO[SPC]
NUMERO:";:IFX=4THENPRINTP:GOTO415
410 PRINTN
415 FORJ=0TO3:FORL=CO+J*40+8*K(N)TOL
+4:POKEL,K(N)+4:NEXTL,J
420 FORJ=0TO3:FORI=S+J*40+8*K(N)TOI+
4:POKEI,160:NEXTI,J
425 FORJ=0TOT1:NEXTJ:POKEWA,16
430 PRINT"[CLR]"B$C$
435 RETURN
449 REM**REPITE LA SECUENCIA********
***
450 FORN=1TONM:PRINT"[CRSRD]REPITE[SPC]
LA[SPC]SECUENCIA":T3=TI
455 GETA$:IFA$=""ANDTI-T3<T2*60THEN4
55
460 IFTI-T3>T2*55THENQ=1:RETURN
```

```
465 Y=VAL(A$):IFY-5<>K(N)THENQ=2:RET
URN
470 IFY-5=K(N)THENGOSUB400:NEXTN
475 IFX=4THENN=P+1
480 IFN=N1+1THENQ=3:RETURN
485 RETURN
499 REM MENSAJES DE ERROR**********
***
500 PRINT"[CLR][2CRSRD]*[SPC]PERDIST
E-NO[SPC]FUISTE[SPC]BASTANTE[SPC]RAP
IDO":GOTO515
510 PRINT"[CLR][2CRSRD]**[SPC]ERROR-
HAS[SPC]PERDIDO"
515 NS=1:FORN=1TO1:NEXTN:IFX=4THENNM
=P
520 PRINT"[CRSRD]"NM-1"PASOS[SPC]FUE
RON[SPC]CORRECTOS":PRINT"[CRSRD][2SPC]
NO.[SPC]DE[SPC]JUEGOS:"G
524 REM**QUINTA SINFONIA************
**
525 FORL=54272TO54296:POKEL,0:NEXT
530 POKEV,15:POKEA,0:POKESR,240:REST
ORE
535 READH:IFH=-1THENFORL=54272TO5429
6:POKEL,0:NEXT:GOTO600
540 READL:READD:POKEHF,H:POKELF,L:PO
KEWA,17:IFNS=1THENPOKEWA,129
545 FORT=1TOD:NEXT:POKEWA,16:IFNS=1T
HENPOKEWA,128
550 FORT=1TO50:NEXT:GOTO535
555 DATA25,177,100,25,177,100,25,177
,100,20,100,800,0,0,250
560 DATA22,227,100,22,227,100,22,227
,100,19,63,800,-1
565 REM**TERMINACION CORRECTA*******
***
570 PRINT"[CLR][2CRSRD]**[SPC]ENHORA
BUENA.[SPC]HAS[SPC]TERMINADO[9SPC][2CRSRD]
[4SPC]CON"N1;"PUNTOS."
575 PRINT"[CRSRD][2SPC]NO.[SPC]DE[SPC]
JUEGOS:"G:TT=TT+N1:PRINT"[CRSRD]**[SPC]
PUNTUACION[SPC]TOTAL:"TT:NS=0
580 GOTO525
599 REM**VER LA ULTIMA SECUENCIA****
***
600 IFX=3ORX=4THEN650
605 IFNM=1AND(X=1ORX=2)THEN650
610 PRINT"[CRSRD]**[SPC]QUIERES[SPC]
VER[SPC]LA[SPC]ULTIMA[16SPC][CRSRD][2SPC]
SECUENCIA[SPC](S/N)?"
615 GETA$:IFA$=""THEN615
620 IFA$<>"S"ANDA$<>"N"THEN615
625 IFA$="S"THEN PRINT"[CLR]"B$C$
630 IFA$="N"THEN650
635 IFN=N1+1THENNM=NM+1
640 FORN=1TONM-1:GOSUB400:NEXTN
649 REM**QUIERES CONTINUAR?*********
***
650 PRINT"[CLR][2CRSRD]*[SPC]QUE[SPC]
QUIERES[SPC]?[28SPC][CRSRD]1.CONTINU
AR[SPC]IGUAL"
655 PRINT"[3SPC][CRSRD]2.CONTINUAR[SPC]
CAMBIANDO[SPC]LAS[SPC]CONDICIONES[3SPC]
[CRSRD]3.TERMINAR."
657 PRINT"[CRSRD]*[SPC]PULSA[SPC]EL[SPC]
NUMERO"
660 GETA$:IFA$=""THEN660
665 Y=VAL(A$):IFY<>1ANDY<>2ANDY<>3TH
EN660
670 ONYGOTO95,675,680
675 GOTO20
680 POKEV,0:POKEWA,0:END
699 REM**FOR STACK CLEARING*********
***
700 IFQ=1THENQ=0:GOTO500
705 IFQ=2THENQ=0:GOTO510
710 IFQ=3THENQ=0:GOTO570
```

Cursillo de Lenguaje Máquina —VI—



Por Diego ROMERO

Vamos al estudio del juego de instrucciones y veremos unas aplicaciones prácticas. Como dijimos en el número anterior, se pueden según su tipo en instrucciones de carga y almacenamiento, transferencia, saltos, lógicas y aritméticas.

Se denominan instrucciones de carga a las que tienen por misión "cargar" algo en un registro de la CPU (LDA, LDX y LDY en todos sus modos de direccionamiento). Las de almacenamiento son aquellas que "almacenan" o guardan el valor contenido en uno de los registros en una posición de memoria (STA, STX y STY). Las de transferencia "transfieren el contenido de un registro a otro (TAX, TAY, TSX, TXA, TXS y TYA). Las de saltos son tres tipos: JMP, JSR y todas las de "branch" BCC, BCS, etc.). Lógicas son aquellas que realizan las "operaciones lógicas" (AND, EOR y ORA), de desplazamiento las que "desplazan" bits de algún dato (ASL, LSR, ROL y ROR), las de comparación "comparan" datos (CMP, CPX y CPY), y aritméticas son las que efectúan este tipo de operaciones (ADC, DEC, DEX, DEY, INC, INX, INY y SBC).

También podemos encontrar otros tipos de instrucciones que son las que afectan a las banderas del registro de estado (CLC, CLD, CLI y CLV), y las que afectan al Stack (PHA, PHP, PLA y PLP).

Para saber qué significan estas siglas sin tener que esperar a que aparezcan en la revista todas las fichas de los mnemónicos, aquí publicamos una tabla del juego de instrucciones en orden alfabético:

ADC	Suma la memoria al acumulador con acarreo.
AND	Hace "AND" con la memoria y el acumulador.
ASL	Desplaza un bit a la izquierda (memoria o acumulador).
BCC	Salta si carry=0.
BCS	Salta si carry=1.
BEQ	Salta si el resultado es cero.
BIT	Compara los bits de la memoria y el acumulador.
BMI	Salta si el resultado es menos.
BNE	Salta si el resultado no es cero.
BPL	Salta si el resultado es más.
BRK	Interrupción por software (break).
BVC	Salta si no hay sobrecarga.
BVS	Salta si hay sobrecarga (V=1).
CLC	Hace C=0 (clear carry).
CLD	Hace D=0 (clear decimal mode).
CLI	I=0 (clear interrupt disable bit).
CLV	Borra bit de sobrecarga (V=0).
CMP	Compara memoria y acumulador.
CPX	Compara memoria e índice X.
CPY	Compara memoria e índice Y.
DEC	Decrementa en uno la memoria.
DEX	Decrementa en uno el índice X.
DEY	Decrementa en uno el índice Y.
EOR	"OR exclusiva" de la memoria con el acumulador.
INC	Incrementa en uno la memoria.
INX	Incrementa en uno el índice X.
INY	Incrementa en uno el índice Y.
JMP	Salta a una nueva dirección (como GOTO en Basic).
JSR	Salta a una nueva dirección guardando la dirección de retorno (como GOSUB).
LDA	Carga el acumulador con la memoria.
LDX	Carga el índice X con la memoria.
LDY	Carga el índice Y con la memoria.
LSR	Desplaza un bit a la derecha (memoria o acumulador).
NOP	No opera (no hace nada).
ORA	"OR" de la memoria con el acumulador.
PHA	Guarda el acumulador en el stack.
PHP	Guarda el registro de estado en el stack.
PLA	Saca el acumulador del stack.
PLP	Saca el registro de estado del stack.
ROL	Gira un bit a la izquierda (memoria o acumulador).
ROR	Gira un bit a la derecha (memoria o acumulador).
RTI	Vuelve (return) de una interrupción.
RTS	Vuelve de subrutina (return).
SBC	Resta la memoria del acumulador.
SEC	Pone carry=1.
SED	Pone modo decimal (D=1).
SEI	Deshabilita las interrupciones (I=1).
STA	Almacena (guarda) el acumulador en la memoria.
STX	Almacena el índice X en la memoria.
STY	Almacena el índice Y en la memoria.
TAX	Transfiere el acumulador al índice X.

▶



EXCURSION POR LA MICROPROGRAMACION

INSTRUCTIONS		INMEDIATE			ABSOLUTE			ZERO PAGE			ACCUM			IMPLIED			(IND, X)			(IND), Y			Z PAGE, X			ABS, X			ABS, Y			RELATIVE			INDIRECT			Z PAGE, Y			PROCESSOR STATUS CODES	
NMEMONIC	OPERATION	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	7 6 5 4 3 2 1 0 / N V . B D I Z C	MNEMONIC
ADC	A+M+C→A (4)(1)	69	2	2	6D	4	3	65	3	2							61	6	2	71	5	2	75	4	2	7D	4	3	79	4	3										N V . . . . Z C	ADC
AND	AΛM→A (1)	29	2	2	2D	4	3	25	3	2							21	6	2	31	5	2	35	4	2	3D	4	3	39	4	3										N . . . . . Z .	AND
ASL	C← [7 0] ←0				0E	6	3	06	5	2	0A	2	1										16	6	2	1E	7	3													N . . . . . Z C	ASL
BCC	BRANCH ON C=0 (2)																															90	2	2							. . . .	BCC
BCS	BRANCH ON C=1 (2)																															B0	2	2							. . . .	BCS
BEQ	BRANCH ON Z=1 (2)																															F0	2	2							. . . .	BEQ
BIT	AΛM				2C	4	3	24	3	2																															M_7 M_6 . . . . Z .	BIT
BMI	BRANCH ON N=1 (2)																															30	2	2							. . . .	BMI
BNE	BRANCH ON Z=0 (2)																															D0	2	2							. . . .	BNE
BPL	BRANCH ON N=0 (2)																															10	2	2							. . . .	BPL
BRK	BREAK (See Fig. 1)													00	7	1																									. . . 1 . 1 . .	BRK
BVC	BRANCH ON V=0 (2)																															50	2	2							. . . .	BVC
BVS	BRANCH ON V=1 (2)																															70	2	2							. . . .	BVS
CLC	0→C													18	2	1																									 0	CLC
CLD	0→D													D8	2	1																									. . . .	CLD
CLI	0→I													58	2	1																									. . . . . 0 . .	CLI
CLV	0→V													B8	2	1																									. 0	CLV
CMP	A-M	C9	2	2	CD	4	3	C5	3	2							C1	6	2	D1	5	2	D5	4	2	DD	4	3	D9	4	3										N . . . . . Z C	CMP
CPX	X-M	E0	2	2	EC	4	3	E4	3	2																															N . . . . . Z C	CPX
CPY	Y-M	C0	2	2	CC	4	3	C4	3	2																															N . . . . . Z C	CPY
DEC	M-1→M				CE	6	3	C6	5	2													D6	6	2	DE	7	3													N . . . . . Z .	DEC
DEX	X-1→X													CA	2	1																									N . . . . . Z .	DEX
DEY	Y-1→Y													88	2	1																									N . . . . . Z .	DEY
EOR	AVM→A (1)	49	2	2	4D	4	3	45	3	2							41	6	2	51	5	2	55	4	2	5D	4	3	59	4	3										N . . . . . Z .	EOR
INC	M+1→M				EE	6	3	E6	5	2													F6	6	2	FE	7	3													N . . . . . Z .	INC
INX	X+1→X													E8	2	1																									N . . . . . Z .	INX
INY	Y+1→Y													C8	2	1																									N . . . . . Z .	INY
JMP	JUMP TO NEW LOC				4C	3	3																												6C	5	3				. . . .	JMP
JSR	JUMP SUB (See Fig. 2)				20	6	3																																		. . . .	JSR
LDA	M→A (1)	A9	2	2	AD	4	3	A5	3	2							A1	6	2	B1	5	2	B5	4	2	BD	4	3	89	4	3										N . . . . . Z .	LDA
LDX	M→X (1)	A2	2	2	AE	4	3	A6	3	2																			BE	4	3							B6	4	2	N . . . . . Z .	LDX
LDY	M→Y (1)	A0	2	2	AC	4	3	A4	3	2													B4	4	2	BC	4	3													N . . . . . Z .	LDY
LSR	0→ [7 0] →C				4E	6	3	46	5	2	4A	2	1										56	6	2	5E	7	3													0 . . . . . Z C	LSR
NOP	NO OPERATION													EA	2	1																									. . . .	NOP
ORA	AVM→A	09	2	2	0D	4	3	05	3	2							01	6	2	11	5	2	15	4	2	1D	4	3	19	4	3										N . . . . . Z .	ORA
PHA	A→Ms S—1→S													48	3	1																									. . . .	PHA
PHP	P→Ms S—1→S													08	3	1																									. . . .	PHP
PLA	S+1→S Ms→A													68	4	1																									N . . . . . Z .	PLA
PLP	S+1→S Ms→P													28	4	1																									(RESTORED)	PLP
ROL	←[7 0]←[C]←				2E	6	3	26	5	2	2A	2	1										36	6	2	3E	7	3													N . . . . . Z C	ROL
ROR	→[C]→[7 0]→				6E	6	3	66	5	2	6A	2	1										76	6	2	7E	7	3													N . . . . . Z C	ROR
RTI	RTRN INT (See Fig. 1)													40	6	1																									(RESTORED)	RTI
RTS	RTRN SUB (See Fig. 2)													60	6	1																									. . . .	RTS
SBC	A-M-C→A (1)	E9	2	2	ED	4	3	E5	3	2							E1	6	2	F1	5	2	F5	4	2	FD	4	3	F9	4	3										N V . . . . Z (3)	SBC
SEC	1→C													38	2	1																									 1	SEC
SED	1→D													F8	2	1																									. . . . 1 . . .	SED
SEI	1→I													78	2	1																									. . . . . 1 . .	SEI
STA	A→M				8D	4	3	85	3	2							S1	6	2	91	6	2	95	4	2	9D	5	3	99	5	3										. . . .	STA
STX	X→M				8E	4	3	86	3	2																												96	4	2	. . . .	STX
STY	Y→M				8C	4	3	84	3	2													94	4	2																. . . .	STY
TAX	A→X													AA	2	1																									N . . . . . Z .	TAX
TAY	A→Y													A8	2	1																									N . . . . . Z .	TAY
TSX	S→X													BA	2	1																									N . . . . . Z .	TSX
TXA	X→A													8A	2	1																									N . . . . . Z .	TXA
TXS	X→S													9A	2	1																									. . . .	TXS
TYA	Y→A													98	2	1																									N . . . . . Z .	TYA

(1) ADD 1 to "N" IF PAGE BOUNDARY IS CROSSED
(2) ADD 1 TO "N" IF BRANCH OCCURS TO SAME PAGE
ADD 2 TO "N" IF BRANCH OCCURS TO DIFFERENT PAGE
(3) CARRY NOT = BORROW
(4) IF IN DECIMAL MODE Z FLAG IS INVALID
ACCUMULATOR MUST BE CHECKED FOR ZERO RESULT

X	INDEX X	+	ADD	M_7	MEMORY BIT 7
Y	INDEX Y	—	SUBTRACT	M_6	MEMORY BIT 6
A	ACCUMULATOR	Λ	AND	n	NO. CICLES
M	MEMORY PER EFFECTIVE ADDRESS	V	OR	#	NO. BYTES
Ms	MEMORY PER STACK POINTER	V	EXCLUSIVE OR		



EXCURSION POR LA MICROPROGRAMACION

- **TAY** Transfiere el acumulador al índice Y.
- **TSX** Transfiere el puntero del stack al índice X.
- **TXA** Transfiere el índice X al acumulador.
- **TXS** Transfiere el índice X al puntero del stack.
- **TYA** Transfiere el índice Y al acumulador.

Puede parecer rara e inútil la instrucción NOP, ya que no hace absolutamente nada, pero esta instrucción es muy útil en contra de lo que se puede suponer. Sus aplicaciones principales son las de introducir un pequeño retardo en algún lugar de un programa, y su función más importante es en el desarrollo de programas, cuando todavía no hemos conseguido una versión definitiva y seguramente se nos olvida alguna instrucción importante, si hemos colocado unas cuantas instrucciones NOP por aquí y por allá, bastará con cambiarlas por las instrucciones que olvidamos, con esto nos ahorraremos tener que reescribir todo el programa.

Quizás alguno de los recién llegados se pregunte cómo en basic puedes intercalar las instrucciones que olvides con sólo escribir otra línea intermedia, y en máquina no se puede hacer esto, la explicación es sencilla: en basic los saltos se hacen a un número de línea, y cuando añadimos o disminuimos algo del número sigue siendo el mismo aunque alteremos la posición de memoria que ocupa. En código máquina los saltos son a direcciones de memoria, y si desplazamos parte del programa al intercalar algo, las instrucciones se moverán de lugar y los saltos no serán los adecuados. La excepción de esta regla son las rutinas reubicables (relocatable), que son aquellas que no contiene este tipo de saltos y que pueden cambiar de dirección sin afectar su correcto funcionamiento.

Vamos a ver ejemplos sencillos de programas en código máquina, el primero de ellos rellena una zona de memoria con un dato.

```
START LDA DATO
      LDX #$FF
LAZO  STA DESTINO,X
      DEX
      BNE LAZO
      RTS
```

Esto en basic sería:

```
10 A=DATO
20 X=255
30 POKE(DESTINO+X), DATO
40 X=X-1
50 IF X <> 0 THEN 30
60 END
```

Este programa podría hacerse más sencillo con un lazo FOR/-NEXT, pero lo he escrito de este modo para que sea idéntico al de código máquina.

Una aclaración, en realidad el programa que aparece aquí no está en código máquina, está en lenguaje ensamblador, y necesitaremos realizar el proceso de ensamblado (pasarlo a c.m.) por alguno de los procedimientos posibles:

El más sencillo pero más caro es utilizar un macro-ensamblador (el C64 dispone de uno muy bueno que permite la utilización de etiquetas, detecta errores, etc.). Para el VIC-20 existen cartuchos como el monitor de lenguaje máquina, que facilita el ensamblado pero no admite la utilización de etiquetas.

Por último se puede realizar el proceso a mano, este es el método más didáctico pero al mismo tiempo es el más pesado e inhumano, consiste en ir buscando las instrucciones una por una en las tablas de mnemónicos, mirar en la columna del modo de direccionamiento que vamos a utilizar y colocar el dato en la memoria utilizando instrucciones POKE después de haber convertido el valor hexadecimal en decimal.

Respecto al modo de utilizar los ensambladores, no lo voy a explicar ya que los manuales respectivos dan todas las instrucciones para su funcionamiento y varían bastante de uno a otro.

Vamos por el camino difícil, lo haremos a mano:

—Primero escogemos una posición de memoria RAM que esté libre. Como la rutina es corta podemos utilizar el buffer de cassete que se encuentra en $033C=828.

—Decidimos que el dato sea $20 y que el destino será $0400 (pantalla en el C64).

—Buscamos LDA en las tablas de mnemónicos y vemos que en modo inmediato es A9, por tanto LDA #$20 será A9 20.

—Seguimos buscando el resto de las instrucciones de igual modo, pero al llegar a BNE LAZO tenemos un problema al hacerlo a mano, debemos calcular la "distancia del salto", para ello restamos la dirección de la instrucción siguiente a la dirección de destino y ese es el salto.

—Cuando terminemos con todo vamos colocando los datos en las posiciones 828 y siguientes, y luego podemos probar la rutina con SYS 828.

Si el ensamblado lo hemos realizado correctamente, debemos borrar la parte alta de la pantalla en un instante. Si queremos borrar menos caracteres podemos cambiar el número $FF en LDX #$FF para que borre lo que queramos.

En código máquina la rutina debe queramos.

```
033C A9 20     LDA#$20
033E A2 FF     LDX#$FF
0340 9D 00 04  STA$0400,X
0343 CA        DEX
0344 D0 FA     BNE$0340
0346 60        RTS
```

Para el VIC-20 debemos cambiar el STA$0400,X, por la dirección donde se encuentre la pantalla, $1E00 si no tiene ampliación RAM, que sería 9D 00. 1E.

La columna de la izquierda del listado es la dirección donde lo colocamos (en hexadecimal). Las del centro son el código de operación y el operando, que es lo que debemos colocar en la memoria para decirle al ordenador que haga "eso", y lo de la derecha son los mnemónicos. Si disponemos de un monitor de lenguaje máquina bastará teclear la columna de la izquierda y las del centro, con un ensamblador entraríamos la de la izquierda sólo la primera vez, y luego la de la derecha, ya que el mismo calcularía la siguiente dirección (columna de la izquierda).

Si tenemos que entrar los datos sin ayudas desde el basic, convertiremos los datos en decimal (169, 32, 162, 255, 157, 0, 4, 202, 208, 250, 96), y los introducimos mediante POKE en modo directo o colocándolos en sentencias DATA y con un lazo FOR, READ, POKE, NEXT, o similar. Como ejemplo, aquí tenéis un cargador para este programa:

```
10 READ A:IF A=-1 THEN
   END
20 POKE 828 + Z, A:Z = Z +
   1:GOTO 10
100 DATA 169, 32, 162, 255,
    157, 0, 4, 202, 208, 250, 96,
    -1.
```

Para probarlo ya sabéis, RUN para colocarlo en sus posiciones y luego SYS828 y RETURN para ejecutarlo.

Si queréis podéis probar su equivalente en basic y comparar los tiempos.

MARKET CLUB

Servicio gratuito para nuestros lectores particulares. Empresas 300 ptas. por línea.
De no especificar duración, los anuncios serán publicados durante 1 mes.
Debido a ciertos abusos que nos han sido comunicados, los anuncios gratuitos de Market club solamente serán publicados si vienen con nombre, apellidos y dirección completa. No se admitirán apartados postales.
Lamentamos que todos tengan que sufrir esta medida por culpa de algunos "*listos*"
Adicionalmente es recomendable que enviéis el número de teléfono por si hubiese alguna pega, si no deseáis que lo publiquemos podéis decirlo en la carta.

MERCADILLO

• Vendo un ordenador Commodore 4000 y otro 3.000. Buen estado. Con programas. Teléfono: (91) 402 38 89 / 401 21 50 ***(Ref. M-70)***

• Vendo VIC-20, Cassete C2N, Placa expansora 4 cartuchos, Curso completo de Basic (2 libros y 4 cintas), Guía de Referencia al Programador VIC-20 y 15 juegos comerciales por 35.000 ptas. (José Lainez. Teléfono: 610 70 80 de 15,30 en adelante) Todo ello comprado en julio de 1982. ***(Ref. M-86)***

• Intercambio programas para C-64 en cassette. Julio Rodríguez, C/Zurbano, 93. 28003 Madrid. Teléf.: (91) 234 98 98. ***(Ref. M-87)***

• Se vende lote de más de 150 programas por 5.000 pesetas, interesados llamar al 236 05 38 ó 231 50 95, o escribir a: C/Mallorca, 420, 6º 1ª. 08013 Barcelona, o a Xavier Querol Carrillo, C/Córcega, 613, 5º 1ª. 08025 Barcelona. Mandamos lista. ***(Ref. M-88)***

• Vendo VIC-20 con 32Kb de RAM, código máquina, 2 cartuchos de juegos, 1 programa de fichero y otro de agenda, curso Basic primera parte, guía de referencia del programador, y mucha información y revistas. Urge vender por cambio de equipo. Juan Altimira Vilar, C/Pintor Togores, 12, 3º 1. Cerdanyola (Barcelona). Teléf.: 93-691 07 28. ***(Ref. M-89).***

• Vendo programa en cinta VIC-20, Ajedrez (nueve niveles); Vic print (procesador texto); Vic calc (hoja cálculo); Graphics (alta resolución); odisea espacial (juegos); Tienes retentiva (educativo); 1ª parte programación + libro todas por 7.000 ptas. Carlos Infante. Tel. (93) 311 27 06 (Barcelona). ***(Ref. M-90)***

• Si alguien ha copiado y vende los programas listados en los dos libros de "curso de introducción al Basic" de Andrew Colin editado por Micro Electrónica, y desea ganarse unas perras dando servicio a los demás, espero que se ponga en contacto conmigo para venderme una copia, ya que quiero introducir a mis hijos con ayuda de estos libros, pero no tengo tiempo. Alfredo Ruiz Martínez, C/Juan Gálvez, 18. 41007 Sevilla ***(Ref. M-91)***

• Vendo un VIC-20, 24.000 ptas. Memoria de 16K 9900 ptas., 5 juegos Indescomp a 800 cada uno, 2 juegos ingleses a 900 cada uno, 1 juego en cartucho 1.100 ptas; 1 libro con 30 juegos, 1.200 ptas.; 1 libro en inglés, 1.200; 50 programas escritos a máquina 1.000, o todo por 36.900. Llamar al 446 65 56 de Madrid. Preguntar por Ismael (a partir de las 16:30) ***(Ref. M-92).***

• Me gustaría cambiar programas y juegos para el VIC-20. Tengo super-expander y cartucho de 16K. Llamad a Jose al 93-422 66 41 (casa) o al 93-332 56 90 (trabajo) Barcelona. ***(Ref. M-93)***

• Cambio controlador doméstico de 8 reles y cartucho de juego ROADRACE por impresora para el VIC y si parece poco también regalo ZX-81 con 16K + juegos con manuales, etc. También cambiaría el ZX-81 con todos sus complementos por super-expander o cartucho de 16K. José Tomás Gómez. C/María Díaz de Haro, 16, 3-F. Bilbao (Vizcaya). ***(Ref. M-94)***

• Desearía intercambiar juegos y todo tipo de programas para el C-64. Escribir a: Eduardo Martín Ruiz, C/Almagro Segura, 4. 18012 Granada, mandando lista de programas, o llamadme al Teléf.: 958-28 22 60. Interesan sobre todo juegos. ***(Ref. M-95)***

• Estoy interesado en el intercambio de programas de juegos para el C-64. Escribir a: Alejandro Canto Navarro, C/Doctor Gadea, 1, 2º E+F, 03003 Alicante o llamar al teléfono 965/ 22 33 30. ***(Ref. M-96).***

• Estoy ¡¡interesadísimo!! en el cambio y compra de juegos y programas de todo tipo del C-64 en cinta. Juan Martínez López, C/Honduras, G-5º, G-9º, 1º, 1ª, A. Barcelona (27). Tel. (93) 340 94 65 ***(Ref. M-97).***

• Desearía intercambiar programas para Commodore 64 con usuarios del mismo. Soy Juan Carlos Stubbs Cruz, C/Mirador, 8 "ALBION" Urbanización "Cerrado de Calderón" 29018 Málaga. Teléfono: 952/29 15 74. ***(Ref. M-98)***

• CBM-64 Intercambio o vendo programas comerciales. Gonzalo Machado Gallas, c/Marín Ocete, 8, 6º F. 18014 Granada. ***(Ref. M-99)***

• Compro programas para C-64 juegos y aplicaciones. En especial ajedrez, trazado de funciones, estadística, etc. Programas de gestión. Todos en cinta o cartucho, Jordi Sierras. Lista de correos, Alella (Barcelona). Tel. 93-555 22 64 ***(Ref. M-100)***

• Cambio cualquier tipo de programas y experiencias para el C-64. Sergi Arbos Mauri, C/Forn del Vidre, 16, 3ª, 4ª, Vilanova i la Geltrú (Barcelona). Tel.: 93-893 18 31 ***(Ref. M-101)***

• Vendo microordenador Dragon 32 por estrenar en perfecto estado, con cables para conectarlo a cualquier cassette y a TV. Regalo 2 juegos en cinta. Manual en castellano. Precio a convenir. Interesados dirigirse a: Eusebio Zuloaga Arisa, C/Balmes 444, 2º 1ª 08022 Barcelona. Tel.: 93-247 22 76 ***(Ref. M-102)***

• Tengo 15 años y el VIC-20, mis posibilidades económicas no me llegan para adquirir un periférico, por esto agradecería si alguien me pudiera vender el Datassette por unas 5.000 ptas. Mis señas son: David Puig Rojals, c/Lorenzana, 36, 4º B. Gerona. Tel. 972-20 46 31. ***(Ref. M-103)***

• Vendo o cambio cartucho de juego ref. VIC-1097 "Jupiter Lander" por cartucho de ampliación Ram de 8K o 3K También Introducción al lenguaje de programación Basic por 3000 ptas. Desearía contactar para intercambio de programas del VIC-20. Francisco Fernández Iglesias, C/Ramiranes, 4, Porrino (Pontevedra) ***(Ref. M-104)***

• Vendo expansor de 4 cartuchos para el VIC-20 marca Indescomp con interruptores individuales por 6.000 ptas y cartucho lenguaje máquina VIC-20 por 3.000. Intercambio programas para C-64 cinta y disco. Miguel Angel Martínez Martínez, C/Madre de Dios, 16, 4º-D, Izda. 26004 Logroño. La Rioja. ***(Ref. M-105).***

• Impresora MPS-801 Commodore vendo por 45.000 ptas. (comprada en mayo 1984) por cambio a modelo mayor. Interesados dirigirse a: Manuel Vijande, Tel.: (985) 25 21 56 de 8 a 10, mañanas y tardes. ***(Ref. M-106)***

• Compraría impresora para CBM64 precio a convenir. ¿Acabas de comprar un C-64 y no sabes cómo usarlo? Vendo la mejor introducción al Basic, sencilla, práctica y fácil de entender: EL CURSO DE INTRODUCCION AL BASIC. PARTE I, de Andrew Colin. Este libro tiene cantidad de ejemplos y programas a probar, muy fáciles de entender. Lo vendo por 1.850 ptas. Mi teléfono es el 229 07 02 de Barcelona. Preguntar por Francesc. Francesc Guasch Ortiz. C/Jerez nº 40. 08032 Barcelona. ***(Ref. M-107)***

• Por cambio de ordenador vendo ampliación de 16K (13.000 ptas.); vídeo-juego en cartucho ALIEN (3.600 ptas.) cartucho de ajedrez SARGON II CHESS (3.600 ptas.); cartucho SUPER EXPANDER (5.800 ptas.) Todo junto por 20.000 ptas. Manuel Fuentes Sorrivas. Apartado de Correos, 1.473. Vigo (Pontevedra). Teléf.: (986) 47 17 68. ***(Ref. M-108)***

• Vendo 3 cassettes de juegos para el VIC-20 por 1.000 ptas. cada uno. Los juegos son: Tron, Skramble y Odisea Espacial (16K). Vendo también la Guía de referencia del programador del VIC-20 y el curso de introducción al basic 1ª parte, cada una por 1.300 ptas. Llamar en horas de comida o cena al teléf.: (985) 56 13 90. O escribir a Benito Martínez Fernández. Colonia del Quirinal, nº 3, 1º izda. Avilés (Asturias) ***(Ref. M-109)***

• Vendo dos programas para el C-64: "Shadowfax" y "Compiler", 3.000 ptas los dos. Agustín Quevedo Velasco, C/Oña, 55, 4º B. Madrid Tel.: 202 94 28 (noches) ***(Ref. M-110)***

• Cambio 2 programas de VIC-20 por 2 de C-64 hechos por equivocación. Son Topo loco y Tron. Más información sobre los programas en Avenida del Norte nº 8 atc. 3ª Manuel González Martínez. Hospitalet (Barcelona). Teléf.: 240 23 50. Llamar sábados por la tarde. Domingos todo el día. Entre semana de 2 a 3 del mediodía ***(Ref. M-111)***

• Deseo ponerme en contacto con socios o colegas radioaficionados para intercambiar programas y experiencias. Tengo un C-64. José Luis Espinar. Indicativo: EA7EBT. Apartado 222. Fuengirola, Málaga. Teléf.: (952) 47 10 76. de 15 a 22 horas ***(Ref. M-112).***

• Vendo VIC-20 con: Superexpander +3Kb, Vicmon (monitor leng. máquina), vic spedd (graba 6 veces más rápido), y más de 150 programas (comecocos, Defensa, Traxx, Vic-calc, Vic-Base, etc.). Francisco Plaza Pérez C/Jesús Rivero Meneses, 2, 47014 Valladolid. Teléf.: 33 23 25. ***(Ref. M-113)***

• Vendo C-64, nuevo, sin uso, comprado en noviembre del 83, con cassette Commodore, comprado en febrero del 84, con libro "guía de referencia" Precio real 94.800, precio de venta: 75.000 ptas. Luis Javier Redondo Fernández; C/Nicasio Pérez núm. 9-3º bis 47005 Valladolid. Teléf.: (983) 29 93 84. (de 11 a 11,30 noches). ***(Ref. M-114).***

• Vendo VIC-20 comprado en enero 1984 por 25.000 ptas. cassette Commodore por 9.000 ptas; 5 juegos de cinta Indescomp y 1 de cartucho Commodore por 14.000 ptas; Guía referencia programador y Basic 1 con cinta por 1.000 ptas. Comprando todo el lote haría más rebaja y regalaría dos colecciones de revistas, joystick y cassette grabado con 11 juegos. Llamen o escriban al teléf.: (91) 613 49 44. Antonio Aguilar Melián. Parque Zarauz Bloque 4º Móstoles (Madrid). ***(Ref. M-115)***

• Vendo y cambio programas para C-64. Base de datos, Ajedrez, Música, juegos, etc. Jordi Bachs, Rambla San Sebastián, 30 4º 1ª, Sta. Coloma de Gramanet, (Barcelona) ***(Ref. M-116).***

• Vendo VIC-20 con expansión de memoria 16K más de 100 programas diferentes y de calidad: Juegos, utilidades, base de datos, etc. cartuchos y específicos para radioaficionados, junto a una extensa documentación y libros. Todo 40.000 ptas. Alfonso Moraleda Pérez. Clara del Rey, 81. 28002 Madrid. Tel.: (91) 415 02 24 - Noches. ***(Ref. M-117).***

• Vendo C-64, unidad disco 154, juego joystic. Apto gestión. Nuevo. Buen precio. Escribir a Ricardo Soteras Sogas, C/Tres Creus, 79. Sabadell (Barcelona) Dar nº Teléfono. ***(Ref. M-118).***

• Desearía contactar con usuarios de C-64, más o menos de mi edad (14 años), para intercambio de programas tanto de juegos como de utilidades (sobre todo de matemáticas). Francisco Javier Almellones, C/Bahía de Algeciras, 8-3º C. Algeciras (Cádiz). Teléf.: 66 49 34 ***(Ref. M-119).***

• Quisiera intercambiar juegos con otros lectores de esta magnífica revista. Mi dirección es: Luis Alfonso Sanz, Pº San Isidro, 5, 2º A. Valladolid-12. ***(Ref. M-120)***

• Vendo Ordenador Personal Commodore 8032. 32K, con Floppy doble 4040, nuevo. Precio total 300.000 ptas. Vendo también PET 2001, 7K, con cassette y pantalla incorporados. Precio 40.000 ptas. Teléf.: 93-674 62 33 (horas de cenar) ó 93/661 55 16 (mañanas). Barcelona ***(Ref. M-121)***

• Por cambio de equipo vendo VIC-20 comprado en noviembre del 83, totalmente nuevo. Regalo al comprador más de 100 programas para VIC básico o con ampliación de memoria (muchos de ellos comerciales y de utilidades, como VIC-PRINT, VIC-CALC, Bases de datos, ajedrez y juegos diversos). Y sólo por

25.000 ptas. Acepto ofertas. Dirigirse a Víctor J. Calvo Medina. Calle Beatas 20, 3º D. 29008 Málaga. Teléf.: 21 44 09. *(Ref. M-122).*
• Vendo VIC-20, casi por estrenar por 39.000 ptas. con cassette manual del usuario, curso Basic 1 (libro y 2 cintas), cartucho de ajedrez, 2 cintas juegos de Indescomp y varias cintas más con 50 programas y juegos. Preguntar por Juan (93) 371 90 03 o escribir al apartado 37 de Esplugues (Barcelona). *(Ref. M-123).*
• Cambiaría toda clase de programas para VIC-20 con o sin superexpander. Tengo algunos muy buenos. También los vendería. Alvaro Ibáñez Blas. C/Cardenal Siliceo, 20, 2º E. Madrid-2. Tfno.: (91) 416 09 13 *(Ref. M-124)*
• Compraría impresora VC-1525 o SEYKOSA, precio a convenir, si es posible que el vendedor sea de la provincia de La Coruña o cercanías también me interesa el monitor de lenguaje máquina. Para ofertas escribir a José Luis Ferreiro Meijide, Apartado 378; La Coruña. *(Ref. M-125)*
• Vendo C-64 por 50.000 ptas. Lote de programas Grandmaster, Base de datos, Gráficos 64, Monopoli, Rox 64, Tank Atack, Hustler, Frogger, Motormania, Flight) todos en 5.000 ptas. Sueltos a 1.000 ptas. Controlador para controlar aparatos externos con 8 relés por 6.000 ptas. Antonio Toral Peinado. C/Puente nº 3 Illora (Granada) Telf.: 958/46 30 09. *(Ref. M-126).*
• Vendo ampliación de 16 RAM para VIC-20 por 12.000 ptas. Razón: Teléf.: 253 09 87. Barcelona. *(Ref. M-127)*
• Vendo programa para C-64. 16 juegos y/o ampliaciones por cinta. 800 ptas. por cinta más gastos de envío. Razón: Maite, C/Moncada nº 5, 2º izda. dcha. Tfno.: (94) 443 67 79. Escribir o llamar. *(Ref. M-128).*
• Compraría programa Assembler (Ensamblador y desensamblador) en cassette para CBM 3000 con instrucciones. Artemio González Pérez-Fernando Arocena Quintero, 1. 38009 Santa Cruz de Tenerife. (Teléf.: 22 42 44). *(Ref. M-129)*
• Vendo NEW BRAIN 32K con display (cables, alimentación, cassette sanyo, manual, guía del principiante, technical software manual, programas). Todo en perfecto estado por 65.000 ptas. Angel Osinaga Rubio. C/José Jimeno 2-2º Drcha. Pamplona 31015 Navarra. Teléf.: (948) 11 35 07. *(Ref. M-130).*
• Busco gente en Sevilla para poder comprar entre bastante gente la guía de programación (que son 2.000 ptas. del C-64) y usar luego el arte de la fotocopia. También busco gente con un nivel medio para compartir programas. Luis Luna Barrios. P/Gomila, 16-10. Sevilla, 7. Tfno.: 954, 51 33 45. *(Ref. M-131).*
• Vendo unidad de disco mod. 1541 con sólo seis meses y muy poco uso por Ptas. 75.000. Interesados escribir a: Alfredo Ruiz Martínez, C/Juan Gálvez, 18. 41007. Sevilla. *(Ref. M-132).*
• Vendo por cambio de ordenador, 15 fabulosos programas para cbm 64: Simons, Basic, Vic Tree, Fort Apocalypse, Choplifter, Solo Flight, Skramble New, Falcon Patrol, Moon Buggy, Pitstop, Soccer 64, Burgertime, Superpipeline, Radar Rat Race, Matrix y Laser Zone por 8.000 ptas. Su precio de mercado es 10 veces. Urgente. Fernando. Horas comida. Tfno.: (91) 415 37 32. *(Ref. M-133).*
• Vendo VIC-20 comprado en dic-83, con el datassete, ampliación 16K, joystick, 5 programas y libros. Todo 45.000 (De no estar de acuerdo con el precio hacer ofertas). Teléf.: (976) 31 01 17 de Zaragoza Javier Boira. Via Ibérica-38, 8º Zaragoza. *(Ref. M-134)*

CLUBS

• Club de amigos. Foto Estudio 2. Plaza de Sombrereros, 2. **Palma de Mallorca.** Tel.: 21 31 62. *(Ref. C-1)*
• Club de programación Alaiz. Tels.: 254 480-257 704. Pamplona. *(Ref. C-4)*
• Club de Commodore de Albacete. Fernando Martínez Guerrero. La Roda, 39. 5º D. Albacete. *(Ref. C-5)*
• Círculo de Durensan Vigués. Vigo. C/Venezuela, 48. Entrechan. Tels.: (986) 410 683/422 519. *(Ref. C-6)*
• Asociación Manchega de Usuarios de CBM-64 (A.M.U. C-64). C/Blasco de Garay, 10. Albacete. *(Ref. C-13).*
• Club MICRO-ESPLAI. Gran Vía, 682. Pral. (Entre Bruch y Gerona) de Barcelona. *(Ref. C-14).*
• Quisiera saber si existe algún Club Commodore en Sevilla. Francisco Javier Recio Lamata. C/Fernán Caballero, 22. Sevilla-1. *(Ref. C-17).*
• Agradecería mucho a todo el que pudiera informarme sobre la creación del Club Commodore en Girona. Daniel Bassas. Gerona capital. Plaza Marqués de Camps, 16, 3º, 2.ª. Tel. 21 27 26. *(Ref. C-18).*
• Club Ensenada de Santander. Club Ensenada. C/Sta. Lucia, 1, 1º. Tel.: 22 59 92. *(Ref. C-19).*
• Estoy formando un club de intercambio de programas (en cinta) para VIC-20. Jaime Vidal Forteza. C/Sor Damiana, 16-1º S. Francisco (Pil. lari). Palma de Mallorca. *(Ref. C-20).*
• Me interesaría conocer conocer si hay algún club de amigos del VIC-20 cerca de mi ciudad. Agustín Ruiz Castrillo. C/Gregorio Solabarrieta, 49, 4º izda. Miranda de Ebro (Burgos). *(Ref. C-21).*
• Estamos formando un Club de Usarios en el centro de Madrid. Cualquier interesado que se ponga en contacto con nosotros en el apartado 427 de Madrid, dirigiéndose a Rafael (EB4XD). Teléf. 276 66 94 (Dejar teléfono. Contestador). *(Ref. C-22).*
• Me gustaría entrar en contacto con usuarios del CBM-64, para intercambio de programas e ideas, así como para trabajar en grupo". Antonio Iglesias Arias. C/San Isidro Labrador, 16, 1º B. Palma de Mallorca, 5. Baleares. Teléf.: 412 728. *(Ref. C-23).*
• Interesados en la formación de un club de usuarios de los ordenadores "VIC-20 y Commodore 64". Escribir o llamar a Félix Portabella Padro. C/Forn de Santa Llucía nº 1, 2º Manresa. Teléf.: 872 22 97. *(Ref. C-25).*
• Agradecería la dirección o teléfono de algún socio de mi ciudad o alrededores, a fin de ponerme en contacto con él para ver la posibilidad de formar un club o similar. José Esteban Llusar Molés. Daoiz, 2-1º 2ª. Pto de Sagunto (Valencia). *(Ref. C-34)*
• Grupo de usuarios de Commodore en Tarragona, estamos intentando formar un club lo más sólido posible, no nos mueve ningún interés lucrativo, y estamos desarrollando un sinfín de actividades, encaminadas a tener un más amplio conocimiento de nuestro ordenador. Alfred Blasi. Tel.: (977) 547 307. Tarragona. *(Ref. C-35).*
• Deseo contactar con niños-as que deseen formar un club de ordenadores Commodore 64 y VIC-20 (puede ser otra marca), para intercambiar experiencias. C/Avda. del Norte nº 8 atc. 3ª Hospitalet (Barcelona). Manuel González Martínez. Tel.: 93-240 23 50. *(Ref. C-36).*
• Interesados en la formación de un club de informática en Granada capital diríjanse por correo a C/ San Jerónimo, 23, 2º. 18001 Granada. Miguel A. Puerta Carrasco. *(Ref. C-28).*
• Si queréis que hagamos un club "Vic-20" poneros en contacto con Antonio Recober. Teléf.: 32 20 46. Málaga. Se trata de intercambiar programas. Conocimientos y experiencia sin que nos cueste dinero. Si algún 64 quiere hacer el camino a nuestro lado. Podemos intentarlo. *(Ref. C-29).*
• Desearía contactar con interesados en formar un club de usuarios del C-64 en Granada e intercambiar programas y experiencias. Antonio Toral Peinado. C/Puente nº 3. Illora (Granada). Tel.: 958-46 30 09 *(Ref. C-30).*
• Para formación club nacional de CBM Series 2000 y 3000 exclusivamente, ponerse en contacto por carta con Artemio González Pérez. Fernando Arocena Quintero, 1. Teléf.: 22 42 44. 38009 Santa Cruz de Tenerife. *(Ref. C-31).*
• Me gustaría contactar con alguien de Motril para compartir experiencias y/o formar un club de C-64 (y VIC-20 si hace falta). Ya tengo local. Mi teléfono: 958-600 314 *(Ref. C-32).*
• Ha nacido el club "Brian & Chip" para usuarios del C-64 y Spectrum 16K. Crearemos una revista propia, pero necesitamos colaboraciones, programas e ideas. Intercambiamos programas listados o en cinta, escribir desde cualquier rincón de España, prometo contestar. Francisco Reig López, C/Marina, 261, 3º 5ª, 08025 Barcelona. *(Ref. C-33).*

(Viene de pág. 4)
que podéis imaginar lo que supone una partícula de polvo en el recinto donde lo fabrican.

La causa de que los chips pongan "made in Japan" es que allí fabrican las cajas al ser más barata la mano de obra, pero lo de dentro es "made in USA". En fin que aprendimos bastante este día. Después la limusine nos llevó al aeropuerto de Filadelfia para tomar el avión de vuelta a Nueva York.

El miércoles por la mañana tomamos otro avión desde el aeropuerto de La Guardia a Boston, donde nos recogió Scott Baum, que es el analista informático de los PCs en la sede de nuestra compañía en Framingham (MA). Nos llevó a comer y fuimos a visitar el edificio de la compañía, conocimos a Marion Kibbee, Debra Wetherbee y todos los jefes de los distintos departamentos, que nos enseñaron cómo atienden a la correspondencia, todo lo hacen con procesadores de texto igual que nosotros utilizamos el easy script, la diferencia estriba en la cantidad de personas que se encargan de la correspondencia y el volumen de esta.

Stott nos llevó de vuelta a Boston, por donde paseamos para ver los puntos interesantes y luego fuimos a cenar en un relajador restaurante situado en uno de los edificios más altos de la ciudad con vistas al puerto. La cena, la compañía y las panorámicas fueron encantadoras.

Al día siguiente llegó Olga Greenwood con el conductor de la limusine, que era similar a la de Commodore pero mejor (además de lo de la otra tenía vídeo para ver películas en el camino..., nos llevó a la sede de la compañía en Peterborough, donde hacen la hermana gemela de la Commodore World en América, que se llama RUN. Allí fuimos a comer con Guy Wright (el editor técnico de RUN), Dennis Brisson (el editor jefe), Matthew Smith y Michele Christian. Olga nos prestó una gran ayuda, ya que hablaba perfectamente español, y puedo asegurar que pasar varios días seguidos traduciendo todas las explicaciones sobre las fábricas de Commodore y circuitos integrados, en las compras y en todo, es una tarea que da bastantes dolores de cabeza, ya que la única persona con la que hablaba mi idioma era con Joan, y para él también fue un alivio poder hablar con otra persona. Luego dimos un paseo en un coche de caballos viendo los frondosos bosques, el lago, y todos los rincones pintorescos incluyendo el rancho donde Michele guarda un potrillo joven. La limusine nos recogió en el rancho para llevarnos a visitar las instalaciones de RUN, 73,s, Hot Co-co, 80 Micro, Incider, etc. Le enseñaron a Joan el proceso de fabricación de la revista desde el principio al fin. Parecían surgir los computadores en todas las fases de la revista, era fantástico. Más tarde volvimos al hotel en Boston y dimos una vuelta por las instalaciones deportivas antes de cenar, Joan prefirió verlo todo, pero a mí me gusta más la natación y aproveché la oportunidad en el hotel, luego cenamos y dimos una vuelta por el Quincy Market que es uno de los lugares típicos de esta ciudad.

El viernes teníamos planeada una visita al acuarium y museos de la ciudad, pero el cansancio hizo que la anulásemos, nos fuimos a dar una pequeña vuelta por la ciudad para hacer las compras de rigor (postales y cosas de esas) y tomamos un avión hacia Nueva York.

Tras todas estas peripecias volvimos al hotel, y el sábado día libre para compras..., también nos recorrimos la parte sur de Manhatan, teníamos que aprovechar el día ya que se terminaba el tiempo y queríamos verlo todo.

El domingo fuimos apenados a tomar el vuelo de vuelta, y unas horas después en casa otra vez, el sueño dorado había terminado y la vida normal se imponía, yo de nuevo a Madrid y Joan a sus montañas pirenaicas. ■

COMENTARIOS COMMODORE

(Viene de pág. 25)
unidades de discos, para recuperar estos datos han añadido un utilitario llamado "regeneración/verificación disco", que se encarga de recuperar los errores producidos si por accidente se destruye parte del disco. También para evitar problemas se han utilizado ficheros de acceso directo del tipo USR, en lugar de los relativos.

De todos estos detalles que he mencionado no tenemos que preocuparnos, ya que a nosotros nos pregunta si queremos copiar el disco, entrar asientos, etc. Todos los detalles técnicos los maneja el programa por sí mismo.

Los Ficheros URS son complicados para el programador, pero evitan errores en los discos y como el programa ya está hecho no nos importa su complicación. El dividir el programa en varios programas, permite no saturar la memoria, con lo que la rutina de "limpieza" se necesita menos veces ganando velocidad y permitiendo que todo el programa sea enorme aunque esté dividido en pequeños bloques. El único aspecto negativo es el pequeño tiempo que se pierde cuando el propio programa carga otra parte, pero creo que merece la pena.

Por último podemos decir que se adapta bien al plan contable español, y yo creo que es un buen paquete de software. ■

PARA LLEGAR MAS LEJOS...

SPECTRAVIDEO™



MKII







MICROSOFT BASIC ★ FORTRAN 80 ★ Microsoft Logo ★ **COLECO cartuchos**
CP/M ★ Cobol 80 ★ MACROENSAMBLADOR Z80 ★ **Ensamblador CP/M**

- MSX. Es el sistema escogido por 16 fabricantes japoneses de ordenadores (SONY, PANASONIC, PIONER, SANYO, SPECTRAVIDEO, etc.) siendo compatibles todos los programas creados por cada uno de los fabricantes.

CARACTERISTICAS GENERALES	SPECTRAVIDEO 328	SPECTRAVIDEO 318	APPLE II e	COMMODORE 64	ATARI 600 xL	BBC MODELO B	DRAGON 32	ORIC ATMOS
PROCESADOR	Z80 A	Z80 A	6502	6510	6502	6502	6809 E	6502
RELOJ EN MHz	4 MHz	4 MHz	1 MHz	1 MHz	1.8 MHz	2 MHz	1 MHz	1 MHz
MEMORIA STANDARD EN RAM	80 K	32 K	64 K	64 K	16 K	32 K	32 K	48 K
AMPLIABLE HASTA	144 K	144 K	–	–	64 K	–	64 K	64 K
MEMORIA STANDARD EN ROM	32 K	32 K	16 K	20 K	24 K	16 K	16 K	16 K
AMPLIABLE HASTA	96 K	96 K	–	–	–	–	–	–
MICROSOFT BASIC EXTENDIDO	SI	SI	–	–	–	NO	SI	SI
NUMERO DE TECLAS	87	51	62	66	57	73	53	57
TECLADO CON GRAFICOS PREDEFINIDOS	SI	SI	–	SI	SI	SI	NO	NO
TECLAS DE FUNCION PROGRAMABLES	10	10	–	8	4	10	NO	NO
MANDO JOYSTICK INCORPORADO	NO	SI	–	NO	NO	NO	NO	NO
COLORES	16	16	16	16	16	16	9	8
SPRITES	32	32	–	8	8	–	?	–
RESOLUCION (Puntos de pantalla)	256×192	256×192	280×192	320×200	320×192	256×640	256×192	240×200
TEXTO EN PANTALLA	40×24	40×24	40×24	40×25	40×24	40×32	32×16	40×28
CANALES DE SONIDO	3	3	1	3	4	1	3	3
OCTAVAS POR CANAL	8	8	4	9	?	3	5	8
A.D.S.R. ENVOLVENTE	SI	SI	NO	SI	NO	SI	NO	SI
CAPACIDAD UNIDAD DE DISCO	256 K	256 K	140 K	170 K	127 K	100 K	?	?
CENTRONICS	SI (opcional)	SI (opcional)	SI (opcional)	–	SI (opcional)	SI (opcional)	SI	SI
RS232	SI (opcional)	SI (opcional)	SI (opcional)	–	SI (opcional)	SI (opcional)	–	–
80 COLUMNAS	SI (opcional)	SI (opcional)	SI (opcional)	SI (opcional)	SI (opcional)	–	–	–
PRECIO P.V.P.	**76.000**	**49.900**	**166.618**	**79.900**	**58.500**	**140.000**	**67.800**	**59.500**

(TM)

MSX y MICROSOFT son marcas registradas por MICROSOFT CORPORATION.
CPM es una marca registrada por DIGITAL RESEARCH INC.
COLECO VISION es una marca registrada por COLECO INDUSTRIES INC.



HARDWARE
DISTRIBUIDOR OFICIAL DE SPECTRAVIDEO INTERNATIONAL LTD.
PASEO DE LA CASTELLANA, 179 - MADRID-16
TELEFONO (91) 279 31 05

DELEGACION EN CATALUÑA: ACE, S.A.
Tarragona, 100 - Barcelona-15.
Teléfono (93) 325 10 58

YA DISPONIBLE EN:



... Y EN TODAS LAS TIENDAS ESPECIALIZADAS.